Rashmi Padiadpu

Title of the Master Thesis

Towards Mobile Learning: A SCORM Player for the Google Android Platform

Keywords

m-learning, Android, SCORM Player, sequential content navigation

Abstract

Current development in mobile technology and the convenience of mobile device us-age have added a new initiative into the learning practice in the form of m-learning. Interoperable content development standards like Sharable Content Object Reference Model (SCORM) make it possible to have a platform independent unit of learning available in small self contained packages for m-learning. SCORM content packages require a continuous connection with the Learning Management System (LMS) to en-able a learning activity. In this thesis, a SCORM player is developed to play recent versions of SCORM content packages, using a run-time environment layer to replace the LMS and to facilitate offl¬ine learning. This SCORM player is implemented on the mobile platform Android and it allows a sequential navigation of the learning content.

Rashmi Padiadpu

Thema der Masterarbeit

Auf demWeg zu Mobile Learning: Ein SCORM-Player fuer die Google-Android-Plattform

Stichworte

m-learning, Android, SCORM-Player, sequentielle Inhalt Navigation

Kurzzusammenfassung

Der gegenwaertige Entwicklungsstand mobiler Technologien und die weite Verbreitung mobiler Geraete haben eine neue Initiative des technologiegestuetzten lernens hervorgebracht, das m-learning. Interoperabilitaetsstandards wie SCORM ermoeglichen den Austausch und die plattformunabhaengige Verbreitung von kleinen, selbstkonsistenten Inhaltsbausteinen fuer den Gebrauch im m-learning. SCORM content packages erfordern kontinuierliche Konnektivitaet zu einem LMS, um den Lernablauf zu steuern. In dieser Arbeit wurde ein SCORM Player konzipiert und entwickelt, der Inhaltspackete aktueller Versionsstandards abspielen kann und eine Laufzeitumgebung bereitstellt, die den LMS-Zutritt fuer ein o¬ffline-Lernen ersetzt. Dieser SCORM Player wurde auf der mobilen Android-Plattform implementiert und unterstuetzt sequentielle Navigation der eLearning Inhalte.

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

Since the advent of mobile devices/phones of the 1980s, mobile devices have become increasingly popular for use in our day-today life without age boundaries. There are estimated to be around 3.3 billion mobile phones in the world today[1] . This is more than three times the number of personal computers (PCs), and today's most sophisticated phones have the processing power higher than a mid-1990s PC. The main contribution to this growth in the number of mobile phone users comes from developing countries. For example, according to one of the main Indian news papers[2], India has over 296.08 million mobile users and is adding over eight million subscribers every month which makes the PC and broadband penetration far below that of mobile phone users. The present generations of school and university students form a considerable part of this mobile device user base. In addition to the steady growth of the mobile industry, an increasing coverage through high-speed networks resulted in not only increased mobile device users but also demands for more services.

The computer-like functionalities such as internet access, multimedia players, document viewers etc offered by mobile devices are leading some observers to speculate that many people in the future will start to see the mobile phone as an alternative to a PC. But some might as well argue that hundreds of millions of people are not going to replace the full screen, mouse and keyboard experience for staring at a little screen. This kind of discussions clearly shows the fact that how powerful and sophisticated mobile devices are becoming.

With the technology advancement, new possibilities are opening up in the education system to access and maintain the knowledge base which contains the on-line course and related learning resources. With the introduction of hypermedia, there is a change in the entire pattern of organization, distribution of learning material in which the traditional teaching methods, books and libraries are augmented by technology supported techniques. With the help of internet, so called e-Learning, a dynamic approach for learning has emerged. e-Learning can be either collaborative or individual. The collaborative approach enables people to learn from one another, e-Learning can connect learners with experts, colleagues, and professional peers with in the scope of the e-Learning system. Still the e-Learning process is individual, because every learner selects activities from a personal menu of learning opportunities most relevant to his/her background. With the individual e-Learning approach, depending on the e-Learning content availability people have opportunities to learn offline or on-line.

Mobile devices are considered as limitations many times, students are inventing ways to use their phones to learn what they want to know. If the educators can use smart ways to deliver the knowledge in a way that fits into today's students enthusiasm, it would leave a bigger and positive impact on the over all learning process. A European study [3] looked at the impact of using mobile devices to aid the learning of a group of young reluctant students and the study showed a positive impact when their learning was supported by mobile devices. The Stanford University initiative to make the lectures available on-line by using iTunes which benefits students globally has been well received with average of 15,000 track downloads per week. Using mobile technologies to engage students in learning hence becomes a very interesting topic of discussion and has been called as m-Learning by experts. M-Learning is unique in a way that it allows a location and time independent personalized learning. It can also be used to enrich, enliven or add variety to conventional lessons or courses by adding quiz, video, audio or animation.

The e-Learning or m-Learning need to maintain the learning contents so that they can be accessible per request over the Internet. In addition, as Von Brevern explains [4]the organization of the contents should enable human like interactive learning experience which depends on many factors like user behavior, how knowledge is represented, built and processed. This requires to look at e-Learning in a cognitive and pedagogical paradigm rather than just as content delivery. To achieve this content need to be reusable and discoverable based on the context. Keeping this in mind to significantly improve the quality and flexibility of learning there are some standards already established by different institutions. SCORM is one among them which could be used to tailor an application to play the knowledge content on a mobile device.

1.2 Goal of This Work

Objectives of an e-Learning environment are to have a broad and easy access to learning contents, provide an interactive and engaged user experience and track the training effectiveness and learning progress. Center of this discussion becomes the e-Learning content which should be small, reusable, sharable and discoverable. So that learning systems can assemble exercise units on a super ordinate level and deliver contents to meet the learner's needs. In the SCORM approach, a Sharable Content Object (SCO) represents the lowest level of content granularity which can be tracked by a LMS. SCOs are self consistent, independent of context, learning resource structures. The goal of this work is to develop and implement a content player to view and navigate the SCOs and to track information that can be communicated between SCO and LMS. Using Android as the development platform, we try to explore the potentials of application development on a mobile device.

1.3 Structure of Chapters

This thesis is organized in 7 chapters. Chapter 2 presents an overview of the e-Learning specifications and standards related to this work. Chapter 3 gives a brief overview of m-Learning, related problems and related work. Chapter 4, describes various platforms available for mobile application development and works related to cross platform application development. Chapter 5 describes the concept of the SCORM Player developed in this thesis. The implementation and testing details of the SCORM Player developed based on the concepts are given in chapter 6. At the end conclusion and future work are given in the Chapter 7.

Chapter 2

e-Learning Specifications and Standards

Since beginning of the concept of e-Learning, there has been a considerable research and development by many organizations. As a result some of them have been successful in establishing their proposals as specifications and standards. According to Webster[1] dictionary "a specification is detailed precise presentation of something or of a plan or proposal for something". When a tentative solution appears to have merit, a detailed written specification is documented so that it can be implemented.Various consortia and collaborations, such as Aviation Industry CBT (Computer-Based Training) Committee[2] (AICC), The Instructional Management Systems (IMS) Global Learning Consortium[3] and Advanced Distributed Learning[4] (ADL) Initiative dedicate teams to focus on documenting the specifications. This specifications document is put to test and systematically tested and reviewed by an accredited standards body and then made globally applicable by removing any specifics of given originators. They are then taken through an open, consensus-based process to produce a working draft. If approved, the working draft receives official certification by the accredited standards body and then it is made publically available to all. There are many institutions and organizations involved in the standardization of e-Learning technologies. e.g. The Learning Technologies Standardization Committee (LTSC) from the Institute of Electrical and Electronics Engineers (IEEE), The 36 subcommittee of the first joint International Standardization Organization and International Electrotechnical Commission Committee[5] (ISO/IEC JTC1 SC36).

The discussion about e-Learning revolves mainly around the terms "learning objects", "learning object meta-data" and "learning object repositories". There

[1]http://www.merriam-webster.com/
[2]www.aicc.org
[3]www.imsglobal.org
[4]www.adlnet.gov
[5]http://www.jtc1sc36.org/

are many definitions for these terms. The IEEE LTSC[6] defines a Learning Object (LO) as "any entity, digital or non-digital, that may be used for learning, education or training". Von Brevern in [4] defined the LOs in a Object Oriented paradigm. i.e. "Objects can be assigned responsibilities, change internal states, communicate by exchanging messages, react to external stimuli, cause effects, and respond to internal causes". There are two major aspects to the learning objects: first, the content of the learning object itself which is the smallest reusable unit of information (audio, video, documents etc), and second, a minimal set of attributes required to adequately describe itself, which could be used for search, access, and reuse of the objects with in the system. To create, organize and manage LOs, XML (eXtensible Markup Language) based approaches have been developed since XML can consistently, unambiguously describe virtually any data using one structure.

The e-Learning environment faces the complex problem of variety starting from the devices used to the learning content. To address this problem the e-Learning standards provide fixed data structures and communication protocols for LOs and e-Learning systems. As mentioned by the ADL technical team [5] the e-Learning standards must address the six "ilities" of e-Learning, namely:

- Interoperability: e-Learning system are no longer a standalone system, but they must have the ability to take instructional components developed in one location with one set of tools or platform and use them in another location with a different set of tools or platform.
- Adaptability: The ability to tailor instruction to provide a flexible workflow and process model to individual and organizational needs. It must support different learning design paradigms such as performance support, training and structured and unstructured forms of learning engagement.
- Re-usability: The flexibility to incorporate instructional components in multiple applications and contexts. It is also critical to save the time and cost of training content development. In a modern e-Learning environment, content must be created in components that are indexed on the basis of standardized meta-data, which allows learning objects or constituent parts to be reused by creators or user of the content.
- Affordability: The ability to increase efficiency and productivity by reducing the time and costs involved in delivering instruction.
- Durability: The ability to withstand technology evolution and changes without costly redesign, reconfiguration or recoding. The e-Learning framework must allow for additional components to be integrated easily using some form of open and component-based software architecture.
- Accessibility: The ability to locate and access instructional components from one remote location and deliver them to many other locations.

[6]http://ieeeltsc.org

When the standards satisfy the above mentioned requirements, then products, course developers and learners can make their selection on quality and appropriateness for their needs rather than compatibility.

According to Fallon and Brown[6] the standards for e-Learning fall into 2 basic categories. First, the Interoperability Standards define how course-ware communicates with administrative systems such as LMS and Learning Content Management Systems (LCMS) to exchange data about learners and their progress. Second, the Content Packaging Standards define how learning objects and group of learning objects, including complete course should be packaged for import into administrative systems, transported between systems and sorted in content repositories so that they can be easily searched for, accessed and reused.

To compare among several available standards we did a short study on the most widely used and the most relevant ones for our need. During the study, it was noticed that most used e-Learning standards are being developed by four main organizations by collaborating with each other. They are :

- AICC is an international group of technology-based training professionals started in early 90's creates CBT related guidelines for the aviation industry. AICC promotes information, guidelines and standards that result in the cost-effective implementation of CBT and WBT(web based training) in the form of AICC Guidelines and Recommendations (AGRs) which references to a detailed specification document.

- IMS Global Learning Consortium is a global, nonprofit organization that strives to enable the growth and impact of learning technology in the education and corporate learning sectors worldwide. It provides a set of specifications related to the development of learning platforms, including specifications for learning evaluation, packaging, and creation of repositories.

- IEEE LTSC is chartered by the IEEE Computer Society Standards Activity Board to develop internationally accredited technical standards, recommended practices, and guides for learning technology.

- ADL Initiative was formed to develop and implement learning technologies across the Department of Defense of the USA.

The specifications and standards discussed here are result of collaborative effort as shown in the figure 2.1. IEEE is one of the accrediting bodies which approves the specifications prepared by AICC, IMS and ADL. ADL initiative is an effort to consolidate the best parts of IEEE, AICC and IMS.

2.1 AICC

AICC specifications being one of the earliest initiative to put together the recommendations for e-Learning, they were built to automate and extend an existing

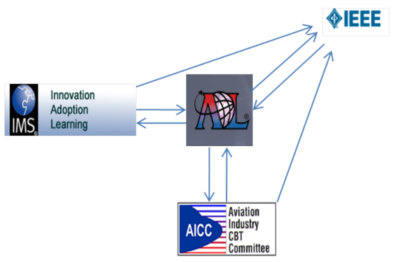
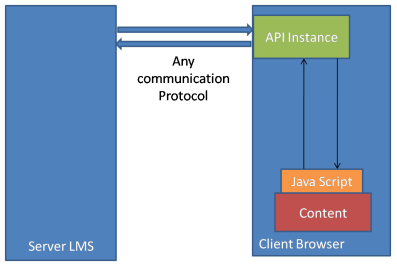


Figure 2.1: e-Learning standards organizations[5]



(a)

Figure 2.2: API communication

process of instructor-led classroom training, rather than to enable a better process that improves learner performance, in addition to being more efficient than the existing process. But they have been updating the specifications to accommodate the new e-Learning concepts with the understanding that learning is not about automating the existing process or to introduce a new channels for delivering content but to deliver the right content, at the right time, over the right channel to the right person. In other words the specifications were updated to address the issue of learner-centricity.

One of the AGRs published so far which is relevant to this work is AICC AGR 010 which references AICC CMI specification(CMI001). This has been standardized by IEEE LTSC as IEEE 1484.11.2-2003 Standard for Learning Technology - ECMAScript Application Programming Interface. This is an interoperability standard which suggests the use of an API Adapter for data exchange between content and administrative systems.

The data that may be communicated between the LMS and learner's client mainly include student, course and progress details. A set of methods were developed to form an Application Program Interface (API) which can be accessed by simple commands in JavaScript as shown in figure 2.2. This provides and easy interface for developers and at the same time allowing LMS to use any desired communication protocol[7]. So use of this method achieves the interoperability of the systems.

2.2 IMS

IMS standards are concerned with both content packaging and establishing interoperability for learning systems, learning content and the enterprise integration of these capabilities. The main specifications from IMS are listed below[8]:

- Simple sequencing : Defines a method for representing the intended behavior of an authored learning experience such that any LMS can sequence discrete learning activities in a consistent way. The specification defines the required behaviors and functionality that conforming systems must implement. It incorporates rules that describe the branching or flow of instruction through content according to the outcomes of a learner's interactions with content.
- Content Packaging (CP): The IMS Content Packaging Information Model describes data structures that are used to provide interoperability of Internet based content with content creation tools, LMS, and run time environments. The objective of the IMS CP Information Model is to define a standardized set of structures that can be used to exchange content. It consists of two major elements, a special XML file describing the content organization and resources in a package, and the file resources being described by the XML. This XML file is called the IMS Manifest file, because course content and organization is described in the context of manifests. XML manifest file has the following elements:
  - references to educational resources included in the package
  - content organization structures of the resources
  - sub-manifest files
  - meta-data for one or more of the elements included in the manifest

  The manifest file and the related resource files are packaged into a single file within an archive format such as zip, jar, cab, tar, etc for transportation, it is called a Package Interchange File (PIF). The PIF provides a concise Web delivery format that can be used to transport content packages between systems.

2.3 IEEE LOM

This is an open standard developed by IEEE for meta-data management in e-Learning resources with the intention of facilitating search, evaluation, acquisition, and use of learning objects, by learners or instructors or automated software processes. LOM is an extension of the Dublin Core Abstract Model (DCAM).

LOM has hierarchy of elements with 9 main categories of meta-data. Each category has an XML binding that has a smallest permitted maximum multiplicity. Each category has subcategories, making of this standard a complex hierarchy of more than 60 different element definitions.

According to IEEE LOM standard, a meta-data instance for a learning object describes relevant characteristics of the learning object to which it applies. Such characteristics may be grouped into categories as described below [9].

- General : groups the general information that describes the learning object as a whole.
- Life-cycle : groups the features related to the history and current state of this learning object and those who have affected this learning object during its evolution.
- Meta-Metadata : groups information about the meta-data instance itself.
- Technical : groups the technical requirements and technical characteristics of the learning object.
- Educational : groups the educational and pedagogic characteristics of the learning object.
- Rights : groups the intellectual property rights and conditions of use for the learning object.
- Relation : groups features that define the relationship between the learning object and other related learning objects.
- Annotation : provides comments on the educational use of the learning object and provides information on when and by whom the comments were created.
- Classification : describes this learning object in relation to a particular classification system.

Above mentioned categories make the LOM V1.0 Base Schema. Describing the components with meta-data provides a common nomenclature and facilitates the search and discovery of the components across systems. According to the IEEE, every LOM meta-data element is optional. It also can be extended by adding new vocabularies to existing elements or adding new elements which gives the flexibility to use this model in verity of applications. The following research groups have used LOM in their implementations: CanCore[7], UK LOM Core[8] and SCORM.

[7]www.cancore.ca

[8]http://zope.cetis.ac.uk/profiles/uklomcore

2.4 SCORM

SCORM (Sharable Content Object Reference Model) is a creation of the ADL Initiative. They describe SCORM as "a collection of standards and specifications adapted from multiple sources to provide a comprehensive suite of e-Learning capabilities that enable interoperability, accessibility and re-usability of Web-based learning content". SCORM specification

- defines a model for packaging learning content to facilitate content delivery
- defines a standard way for e-Learning systems to communicate with each other
- defines a method with the help of meta-data to describe the course content, facilitating efficient and effective identification of appropriate course materials

By conforming to the SCORM standard, different systems can exchange, and reuse course materials. This is especially important for a large-scale distributed learning environment, consisting of multiple dispersed learning management systems managing various learning resources.

Basically there are 3 parts in the latest version which is called SCORM 2004, the Content Aggregation Model(CAM), the Run-Time Environment (RTE) and Sequencing and Navigation(S&N).

2.4.1 The Content Aggregation Model

The Content Aggregation Model specifies how one should package the content so that it can be imported into an LMS and it represents a learning taxonomy neutral means of instruction to aggregate learning resources for the purpose of delivering a desired learning experience. A learning resource is any representation of information that is used in a learning experience. Learning experience consist of activities which involve the creation, discovery and gathering together, or aggregation, of simple assets into more complex learning resources and then organizing the resources into a predefined sequence of delivery. The SCORM CAM supports this process and is made up of the following three parts: the Content Model, the Meta-data and Content Packaging. These three parts have evolved from the basic ideas of the AICC specification that defines the reusable properties and the structural content of learning objects, the IEEE LOM standard meta-data definition and the IMS Content Packaging Specification respectively.

Content Model

This is the nomenclature defining the content components of a learning object. If the content contains more than one module, the content model describes any relationships between those modules called Aggregations. The content model

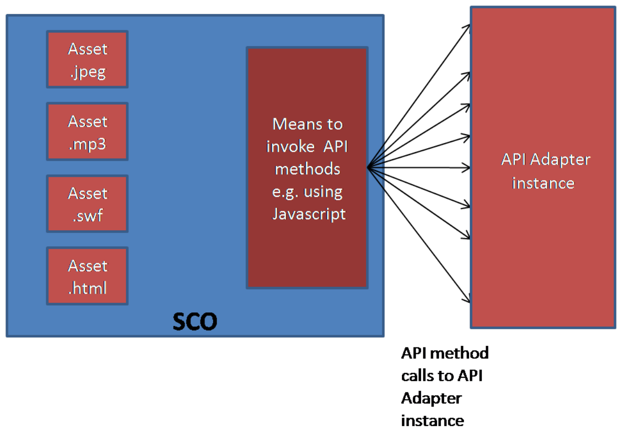

Asset
.jpeg
Asset
.mp3
Asset
.swf
Asset
.html
Means to invoke API methods e.g. using Javascript
SCO
API Adapter instance
API method calls to API Adapter instance

Figure 2.3: A Sharable Content Object

also describes the physical structure of the content and it defines a powerful model for breaking content into arbitrarily sized units of reuse. These units are called Assets and Sharable Content Objects (SCOs).

An Asset is the basic building block of a learning resource. It is defined as an "electronic representation of media, text, images, sound, web pages, assessment objects or other pieces of data or any other piece of data that can be rendered by a web client and presented to a learner"[5]. Examples of Assets include images, sound clips, Flash movies, HTML fragments etc. More than one asset can be collected together to build other assets and an asset can be described with meta-data to allow for search and discovery within repositories, thereby enabling opportunities for reuse and facilitating maintenance.

A SCO is defined as collection of one or more assets that represent a single launchable learning resource that uses the SCORM Run Time Environment (described in 2.4.2) and an API Adapter to communicate with an LMS as shown in figure 2.3. Each SCO should be reusable.Therefore a SCO should not be context sensitive, it should not reference or link to other SCOs. SCOs are intended to be subjectively small units, such that potential reuse across multiple learning contexts is feasible and it represents the lowest level of granularity of a learning resource which could be tracked by the LMS using the SCORM RTE. Similar to assets, a SCO also can be described with meta-data to allow for search and discovery within repositories, thereby enabling opportunities for reuse. A SCO needs to communicate with the LMS when launched, it is required to adhere to the requirements defined in the SCORM RTE. i.e. Every SCO must have a means to locate an LMS provided API Adapter instance and they must invoke at least 2 API methods (Initialize(“”) and Terminate(“”)) (API methods are discussed in next section).

In the content model a learning activity is described as a meaningful unit of instruction; it is something the learner does while progressing through instruction. A learning activity may provide a learning resource (SCO or Asset) to the learner or it may be composed of several sub-activities. Sequencing applies to Activities and it is the structuring of Activities in relation to one another and by associating sequencing information with each Activity. The LMS is responsible for interpreting the sequencing information described in the Content Organization and applying sequencing behaviors to control the actual sequence of the learning resources at run-time.

Meta-data

The Meta-data specification provides a mechanism to describe the content using a pre-defined and common vocabulary. This vocabulary is broken into nine categories as described in IEEE LOM. Even though the Meta-data specification defines a very rich data model, only a small subset of the data elements are required to achieve SCORM conformance.

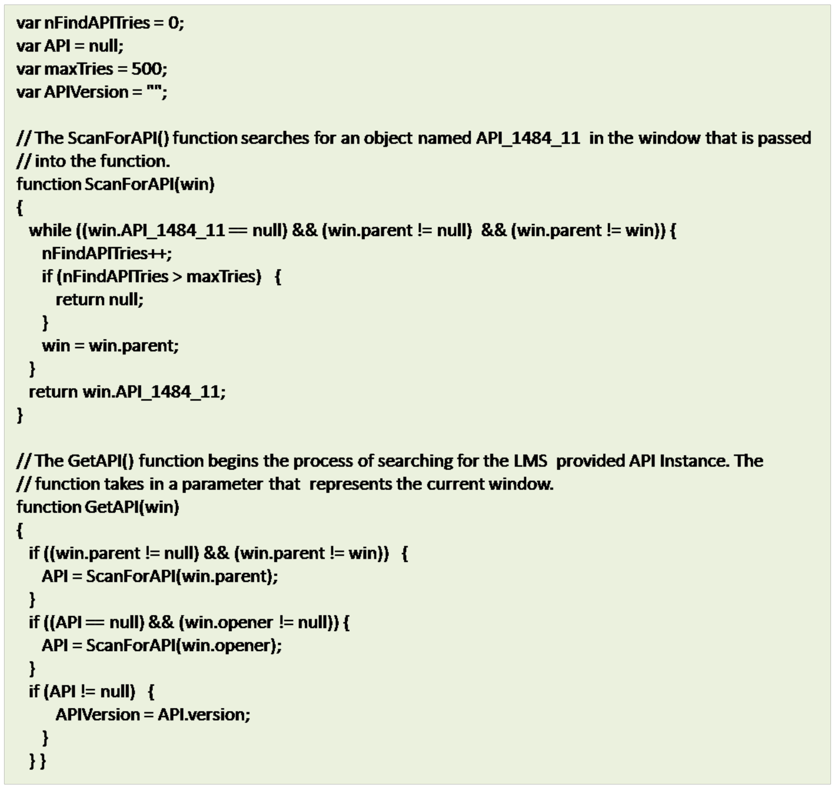
Content Package

SCORM Content Packages adheres strictly to the IMS Content Packaging Specification and provides additional explicit requirements and implementation guidance for packaging Assets, SCOs and Content Organization.An Asset resource or a SCO resource can be comprised of a single or multiple files. The SCORM Resource Content Package Application Profile allows for packaging Assets and SCOs comprised of single or multiple files. Also, Assets and SCOs may be included locally in the package or may be referenced externally. Locally packaged files will be included as physical files within the overall package. When externally referenced, the Assets and SCOs will not be included as physical files within the package, but will instead be referenced by an URL.

2.4.2 The Run-Time Environment

Learning Content Launch

The learning resources in the form of a content package are launch able to the learner's browser. There are 2 types of learning object launching according to SCORM, based on the nature of LOs. i.e. Assets launch and SCO launch. For LO that has only Assets, the only requirement by the SCORM launch model is that LMS must launch the Asset using the HTTP protocol and since an Asset does not communicate to the LMS, there is no tracking for assets via the API and RTE Data Model. For the SCOs, the SCORM launch model requires that an LMS launch and track one SCO at a time per learner. Together with the API instance the LMS launches the SCO in a web browser window of the learner. The API instance should be exposed to the browser as DOM object with a mandatory name given by the IEEE standard [10] as API_1484_11. The launched SCO has to find the API to start the communication process with the LMS. To standardize the process of searching the API instance, the IEEE standard has provided a simple ECMAScript (Figure 2.4) that will find the



```
var nFindAPITries = 0;
var API = null;
var maxTries = 500;
var APIVersion = "";

// The ScanForAPI() function searches for an object named API_1484_11  in the window that is passed
// into the function.
function ScanForAPI(win)
{
  while ((win.API_1484_11 == null) && (win.parent != null)  && (win.parent != win)) {
    nFindAPITries++;
    if (nFindAPITries > maxTries)   {
      return null;
    }
    win = win.parent;
  }
  return win.API_1484_11;
}

// The GetAPI() function begins the process of searching for the LMS  provided API Instance. The
// function takes in a parameter that  represents the current window.
function GetAPI(win)
{
  if ((win.parent != null) && (win.parent != win))   {
    API = ScanForAPI(win.parent);
  }
  if ((API == null) && (win.opener != null)) {
    API = ScanForAPI(win.opener);
  }
  if (API != null)   {
      APIVersion = API.version;
    }
  } }
```

Figure 2.4: API Script

API instance recursively. This ECMAScript contains 2 functions, GetAPI and ScanForAPI. These two functions search for the API in the following locations :

1. The chain of parents of the current window, if any exist, until the top of the window of the parent chain is reached.
2. The opener window, if any.
3. The chain of parents of the opener window, if any exist, until the top window of the parent chain is reached.

Once the API instance is found, the session method "Initialize()" is called to start the communication session between the SCO and the LMS. If the LMS returns a "false" or any value other than "true" back to the SCO, then the SCO can not communicate with the LMS and the learning experience ends immediately. If the method returns a characterstring "true" from the LMS, the communication session is successful and the learner's client continues to provide the learning experience.

SCORM RTE Data Model

During the learning experience, the RTE Data Model layer monitors the learners interaction with the content. In the SCORM player, to make the learning content navigation possible independent of a connection to a LMS, a RTE Data Model layer needs to be implemented. So that every GetValue() and SetValue() method calls from the SCO a element value is retrieved or added or updated in the RTE Data Model with in the mobile device. The RTE Data Model uses a dot notation with every element name starting with "cmi". When a data model element is used in a SCO, they should be used as an ECMAScript character-string and they should be in conformance with the Unicode Standard[11]. The design of the SCO does not impose any mandatory conditions on using any API methods other than initialize and finalize, hence the tracking becomes optional. Therefore the use of data model elements by SCO are optional too. But when they are used, they must follow the SCORM Data Model to ensure their compatibly and re-usability with different LMSs. However, a run-time environment must implement the complete data model elements in case a SCO needs to use any of them.

The RTE data model consists of several data elements. Some of the data elements hold one record per SCO, while some can collect multiple of records with sub elements while tracking a SCO - learner interaction. They are called Collections. RTE Data Model implementation must provide support to handle at least 250 entries per SCO for each collection elements. RTE Data Model contains several categories of data elements as listed below:

- Status data
  - Completion Status - This is a mandatory element for the RTE Data Model and it indicates whether the learner has completed the SCO. There are four status values which could be bound to this element: completed, incomplete, not attempted and unknown. This element influences the sequencing of the learning experience by operating in combination with the elements completion threshold and progress measure. The usage of this element depends on the learning content designer. For example, the completion status can be determined by the number of pages visited by the learner or completion various objectives in the SCO.
  - Success Status - Indicates whether the learner has mastered the SCO.
- Score data - Identifies the learner's score for the SCO
- Mode - Identifies the modes in which the SCO may be presented to the learner. This element is a mandatory and it can be bound with 3 different values: "browse", "normal" and "review".
- Threshold data

  - Time Limit Action - Indicates what the SCO should do when the maximum time allowed is exceeded.
  - Scaled Passing Score - Identifies the scaled passing score for a SCO
  - Maximum Time Allowed - Indicates the amount of accumulated time the learner is allowed to use a SCO in the learner attempt
  - Completion Threshold - This is a mandatory element which identifies a value against which the measure of the progress the learner has made toward completing the SCO

- Data about content objectives and their status - This collection element specifies learning or performance objectives associated with a SCO. Using this element the content designer can set a list of unique objectives by associating each with an identifier. The data collected by the element score, success status, completion status and progress measure are used to evaluate the learner's final performance against the set objectives. During the learning experience, selection of different objectives results in different sequencing experiences. Each set of objective status information consists of the following sub elements: identifier, score, success status, completion status, progress measure and description.

- Data about various types of interactions - This element is a collection which defines information pertaining to interactions between the user and the SCO for the purpose of assessment. During a learning experience every user response is recorded in this element. This element has 10 sub elements: identifier, type, objectives, timestamp, correct responses, weighting, learner response, result, latency and description.

- Comments
  - Comments From Learner - Based on the learning content design, the user can have option to enter his/her comments in the form of texts during the learning experience. This element has sub elements to capture the comment, time and location of the comments made within the SCO.
  - Comments From LMS - This is also a collection element which contains comments and annotations intended to be made available to all learners of the SCO from the LMS. This element has three sub elements - comment, time and location of the comments.

- Learner information
  - Learner Id - Identifies the learner on behalf of whom the SCO instance was launched.
  - Learner Name - Represents the name of the learner.
  - Learner Preference - Specifies learner preferences associated with the learner's use of the SCO.

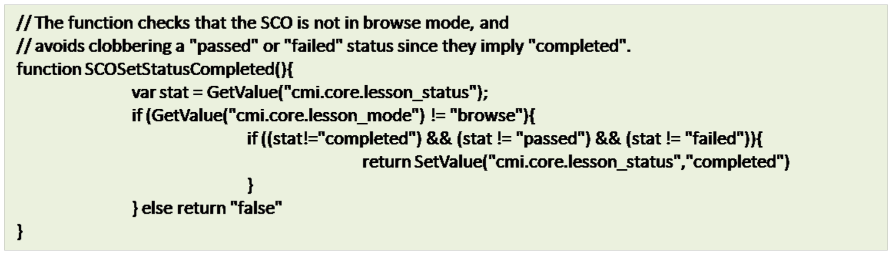
- Suspend data and location, which can be used to resume an interrupted session.
- Entry and exit status, used to determine how the SCO was launched and how the SCO believes it is being terminated.
  - Entry - Contains information that asserts whether the learner has previously accessed the SCO. It can have the value "ab-initio" or "resume" or "". An entry value of "ab-initio" indicates that the SCO has a default set of run-time data which means that there is no run-time data available from any previous learner attempts. An entry value of "resume" indicates that the SCO is accessing run- time data for the current learner attempt as set from the previous learner session on the SCO. An empty entry string can have different meaning as per the content designer's choice.
  - Exit - Indicates how or why the learner left the SCO. A value of "suspended" means that, the SCO can be relaunched for resuming the learning experience. If the values are "normal", "logout","timeout" or "", then the SCO learning attempt ends. Entry and Exit elements are used together while launching a SCO to the learner's browser to start the SCO either from initial state or from the previous session.

In addition, SCORM defines a set of 3 keywords _version, _count and _children which are read only elements used by the SCOs to find the descriptive information about the data elements:

_version	to determine the version of the data model used
_count	is used to determine the number of data elements in a collection
_children	is used to determine all the data model elements in a parent data model element. For example, the cmi.objectives._children data model element represents a listing of supported data model elements that are supported by the LMS. A *GetValue(cmi.objectives._children)* would return a comma separated list like "id, score, success_status, completion_status, progress_measure, description"

The use of RTE data model elements is left to the SCO creator based on the details which needs to be tracked during the course of the learning experience. An example for the use of the RTE data model elements *cmi.core.lesson_mode* and *cmi.core.lesson_status* by the SCO is as shown in figure 2.5. Here the JavaScript function checks if the learning content is not in browse mode and then based on the current lesson status, sets the SCO status to completed.

The collection elements like interactions or objectives are always used with their sub elements. For example, consider the element *interaction* which has the collection *objective* as its sub element. To access the mthobjective's identifier associated with the nthinteraction the parameter to the GetValue is *cmi.interactions*



```
//The function checks that the SCO is not in browse mode, and
// avoids clobbering a "passed" or "failed" status since they imply "completed".
function SCOSetStatusCompleted(){
                var stat = GetValue("cmi.core.lesson_status");
                if (GetValue("cmi.core.lesson_mode") != "browse"){
                                if ((stat!="completed") && (stat != "passed") && (stat != "failed")){
                                                return SetValue("cmi.core.lesson_status","completed")
                                }
                } else return "false"
}
```

Figure 2.5: Sample Script

.n.objectives.m.id. In a SCO JavaScript to get the values from RTE Data Model it could be used as *GetValue("cmi.interactions." + n + ".objectives." + m + ".id")*. The implementation of the relationship between the data model elements is implementation specific.

The Run-Time Environment has evolved from the AICC's API communication method. In addition it also specifies how the content should behave once it has been launched by the LMS. A SCORM conformant LMS is required to implement an API consisting of eight functions [5] which the content may access to communicate with the LMS. The communication is initiated in one direction, from the SCO to the LMS. The SCO always invokes functions on the LMS's API Instance. The LMS does not invoke any functions defined by the SCO other than responding to the call initiated by the SCO. SCORM does not place any restrictions on the underlying communication infrastructure of the API Instance and SCORM requires the LMS to provide an instance of the API as defined by the IEEE standard but the actual implementation is out of the scope for SCORM.

This API is implemented by an API Adapter. An API Adapter must reside in a window that is an opener window or a parent frame of the window that contains the content. This means that the LMS may launch the content either in a new window or in a frame set. The API Adapter must be an ECMAScript (JavaScript) object named "API" that is accessible though the DOM (Document Object Model). All communication between the content and the LMS is handled by this adapter, thus the content author does not need to worry about communicating with the server, but only needs to be able to find the API Adapter and make the appropriate JavaScript calls. This separation of client and server is essential to SCORM so that it ensures the portability of content by forcing it to run on a standard platform which is normally the web browser. It is important to note that content can only communicate with the LMS through this JavaScript API Adapter. There is no other SCORM conformant methods for content to communicate with the LMS such as web services or HTTP requests.

The functions needs to be implemented with in the API Adapter are categorized as session, data transfer and support methods as shown in figure2.6. For minimal SCORM conformance, the only thing that a piece of content needs to

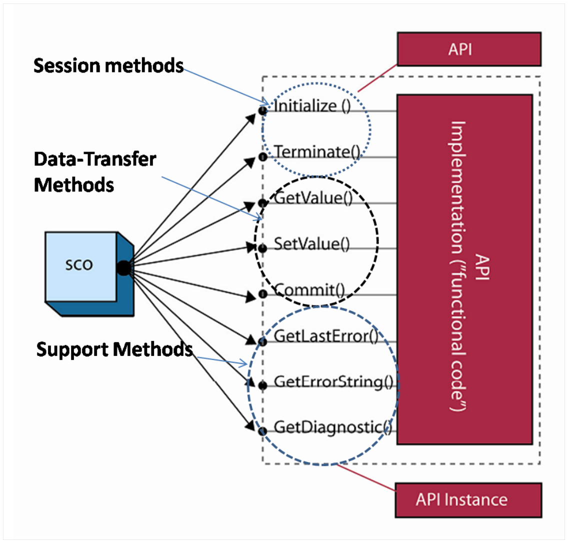


Figure 2.6: SCORM Run Time Environment API

do is call the session methods, Initialize() when it starts and then call Finish() when it exits. Session methods are used to mark the beginning and end of a communication session between a SCO and an LMS. These methods return true or false to the SCO based on the success or failure of communication session initialization and termination. As a result SCO reach one of these states:

- Not Initialized (SCO is just launched, not started)
- Running (Session established with the LMS)
- Terminated(End of session)

In the real-world, the learning process needs higher interactions and learner wants to be able to report test results, track time, bookmark the last location etc. Once the SCO reaches "running" state, the next three methods are used, to exchange data model values between a SCO and the LMS. They are called as data transfer methods. The SCORM defines a data model consisting of data model elements which the content can read from and write to, facilitating this kind of functionality. On occurrence of an error, error code defined by the IEEE standard[10] is set which are integers represented as character strings.

- GetValue(parameter) retrieves a data model element's value from the LMS where the parameter represents the complete identification of a data model element. The method can return one of two values either a character string containing the value associated with the parameter or an empty string if an error occurs and it also sets an error code to a value specific to the error.

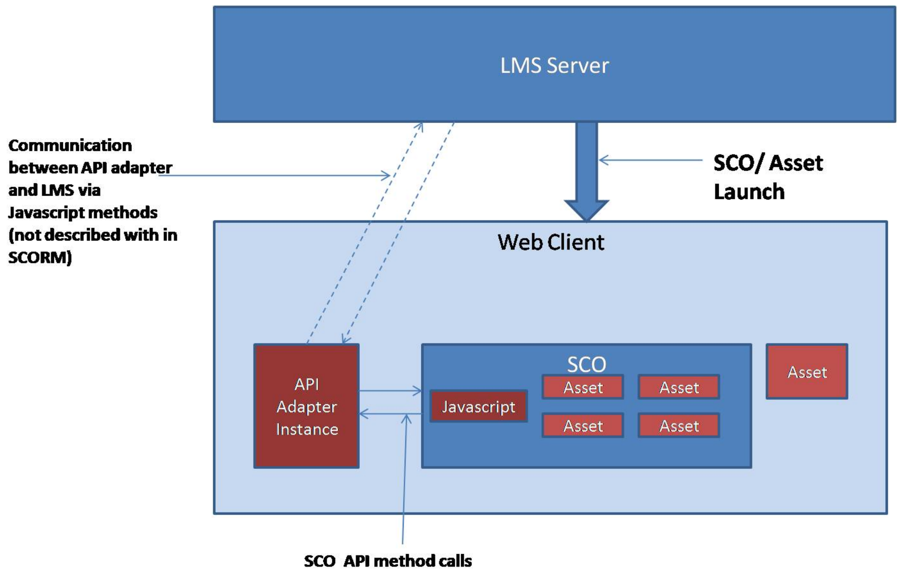


Figure 2.7: SCORM RTE

- SetValue(parameter1, parameter2), This method allows the SCO to send information to the LMS for storage, where parameter1 is the complete identification of a data model element and parameter2 is the the value to which the contents of parameter1 is to be set. The method can return one of two values either true or false where true indicates the successful operation and false indicates the failure. The failed operation also sets the error code to a value specific to the error.
- Commit() method requests forwarding to the persistent data store any data from the SCO that may have been cached by the API Instance since the last call to Initialize or Commit whichever occurred most recently. The method can return one of two values either true or false where the value true indicates the successful operation and false indicates the failure. The failed operation also sets the error code to a value specific to the error.

Example:

```
cmi.core.lesson_location
```

is the data element that describes the user's location in the content. When the content begins (after it has called Initialize();), it may want to make this call to find out where the user left off and return him to that point:

```
strLastLocation = objAPI.GetValue ("cmi.core.
    lesson_location");
```

When the content goes to another area, it might make this call to save the user's location:

```
blnSuccess = objAPI.SetValue("cmi.core.lesson_location",
    "page5");
blnSuccess = objAPI.Commit("");
```

The support methods allow the SCO to trap and intelligently deal with errors and give diagnostic information. With each API function described so far, and only those described error conditions may occur. When these error conditions are encountered the error code is changed to indicate the error encountered. Calling any of the support methods shall not change the current error code. These support methods allow the SCO to determine if an error occurred and how to handle any error conditions encountered.

Implementing this API Adapter in the LMS is a little more involved than using it from content. The API Adapter has to implement all of the API functions and support most of the SCORM data model. The issue involved with implementing a SCORM conformant LMS is how to handle the browser-to-server communication. Some commercial SCORM course developers have chosen to do this with a Java applet and some have been successful using Flash, ActiveX controls and pure JavaScript. A typical implementation of the SCORM RTE should look like as shown in figure 2.7. In this model the LMS server acts as the SCO repository and has the capability to launch a SCO or an asset to the web client (normally a web browser). It also provides an instance of API adapter to the web client. Using the API methods available in the API Adapter, SCO communicates with the LMS. Assets do not contain any communication means and hence can just be played with in the web client.

2.4.3 Sequencing and Navigation Specification

This specification is based on the IMS Sequencing and Navigation Specification.

A learner experiencing SCORM content does not necessarily exercise complete control over which learning object they experience and when. SCORM Sequencing and Navigation define the ability of a learner to navigate from one learning object to another and the sequence in which learning objects may be experienced by a learner. Content developers determine how much control is given to the learner based on organization and sequencing information defined in the content package's manifest. When accessing SCORM content, a learner will experience only one learning object at a time. In SCORM, a learning object is not allowed to explicitly link to another. Rather, when a learner wishes to experience an activity they use navigation controls provided by the LMS (e.g. Start, Continue, Quit, etc) to issue navigation requests. The navigation requests are processed by the LMS to determine the sequence of learning activities[5].

SCORM Sequencing depends on the following main concepts:

- an activity tree representation of learning activities
- the Sequencing Definition Model
- the Sequencing Behaviors

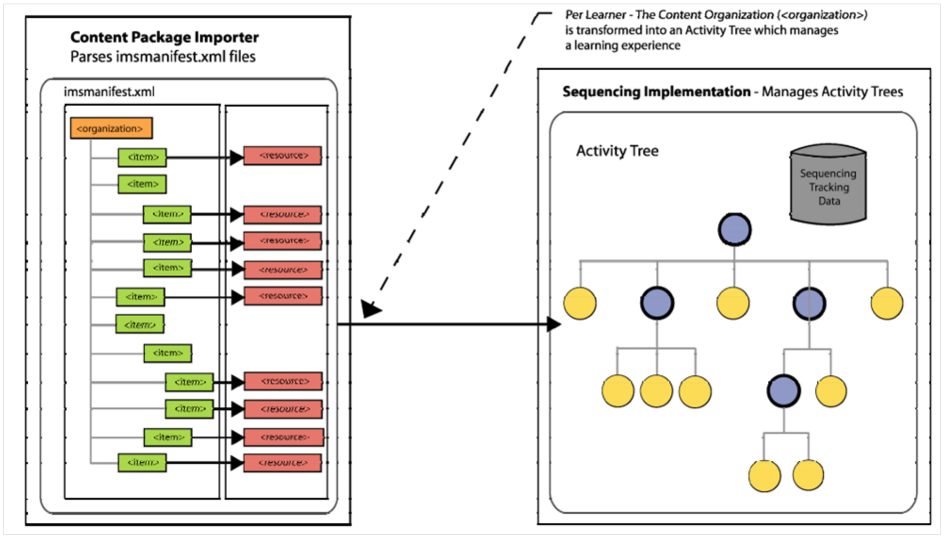

Content Package Importer
Parses imsmanifest.xml files
imsmanifest.xml
<organization>
<item>
<resource>
Per Learner - The Content Organization (<organization>) is transformed into an Activity Tree which manages a learning experience
Sequencing Implementation - Manages Activity Trees
Activity Tree
Sequencing Tracking Data

Figure 2.8: SCORM Activity tree [5]

Activity Tree

This is a run-time representation of the <organization> structure defined in a content package's manifest file. The <organization> element is the root of the activity tree and each of its <item> elements corresponds to a learning activity. Each activity in an activity tree represents an instructionally relevant unit of learning. An activity is a unit of instruction that may consist of a launch able learning object (leaf) or be composed of other learning activities (cluster). An example is shown in the figure 2.8

Sequencing Definition Model

The SCORM Sequencing Definition Model defines a set of elements that may be used to define intended sequencing behavior. These elements are applied to learning activities within the context of an Activity Tree. Each element has a default value that is to be assumed by any sequencing implementation in the absence of an explicitly defined value. The effects of the SCORM Sequencing Definition Model elements only apply during the application of SCORM Sequencing Behaviors. A SCORM conformant LMS must support the behaviors that result from the values associated with all of the defined Sequencing Definition Model elements.

Sequencing Behaviors

Here the SCORM conformant LMS maintains 2 models for individual activity and the activity tree as a whole. First one is a dynamic run-time data model called Tracking model which captures information gathered from a learner's interaction with the content objects associated with activities while the learner is interacting with a content object and the LMS. The second one is called Activity State model which manages sequencing state of each activity in the Activity Tree and the global state of the Activity Tree. This is a dynamic run-time data model utilized by the LMS's sequencing implementation to manage the state of the Activity Tree during a sequencing session.

SCORM Navigation defines how learning and system initiated navigation events are triggered and processed, resulting in the identification of learning activity for delivery. For a learner to access a course or any of its activities, it must issue a navigation request. The result of each navigation request is that an activity is delivered to the learner and the current activity is taken away. The delivery of the activity in response to a navigation request is defined by the content package's activity tree and sequencing information. There are a list of navigation requests described in SCORM, they are:

- Start - request to identify the first or starting activity of a tree
- Resume All - request to resume a previously suspended attempt on an activity tree
- Continue - request to identify the "next" logical learning activity in relation to the current activity available in the tree
- Previous - request to identify the "previous" logical learning activity in relation to the current activity in the tree
- Choose - request to "jump" directly to a specific learning activity in the tree
- Abandon - request to prematurely or abnormally terminate the current attempt on the current activity
- Abandon All - request to prematurely or abnormally terminate the current attempt on the root activity of the tree
- Suspend All - request to "pause" the current attempt on the root activity of the tree
- Unqualified Exit - indicates the attempt on the current activity has finished normally and not as the result of another navigation event
- Exit All - indicates the current attempt on the root activity of the tree has finished normally

2.4.4 SCORM Versions

The original version of SCORM, version 1.0, was a proof of concept only. It introduced the notion of Sharable Content Objects (SCOs) and the API model in which the burden of managing communication across the Internet is handled by the run-time environment, not by the content objects. The first production version of SCORM was version 1.1. It used a Course Structure Format XML file based on the AICC specifications to describe content structure, but lacked a robust packaging manifest and support for meta-data. Version 1.1 was quickly replaced by SCORM 1.2. SCORM1.2 was the first version with a real conformance test in the form of a test suite[12]. It uses the IMS Content Packaging specification with full content manifest and support for meta-data describing the course. Also allows optional detailed meta-data tagging of the content objects and assets described in the manifest. Version 1.2 is no longer maintained or supported by ADL.

The current version of SCORM is version 1.3, also known as SCORM 2004. It includes the ability to specify adaptive sequencing of activities that use the content objects, new standards for API communication. SCORM 2004 also includes the ability to share and use information about success status for multiple learning objectives or competencies across content objects and across courses for the same learner within the same learning management system.

The key changes from SCORM Version 1.2 to SCORM 2004 are as follows :

- The addition of learning content sequencing and navigation capabilities as defined by the IMS Global Learning Consortium's Simple Sequencing (SS) specification to address the need for dynamic presentation of learning content based on learner performance.

- Synchronization of the Content Packaging Specification with the maintenance updates by IMS on the Content Packaging Specification Version 1.1.4.

- Refinements and bug fixes in the other e-Learning interoperability specifications that are part of the SCORM. i.e the IEEE's ECMAScript Application Programming Interface (API) used in the SCORM RTE and IEEE Learning Object Metadata (LOM) used in the SCORM CAM became formal standards. According to these standards SCORM RTE and CAM were synchronized.

- IEEE Data Model for Content Object Communication and XML Schema Binding for LOM has been added to SCORM.

Chapter 3

M-Learning

3.1 Problem Statement

The studies conducted so far have reported that use of mobile technologies in learning have given very encouraging results. In Japan a project called 'Pocket Eijiro'[1] conducted in year 2002 has shown that m-learners have quite smaller dropout rate than e-learners. Another ongoing project (since 2003) called Learning2Go[2] in Wolverhampton, UK, has shown that usage of the handheld devices in schools have positive impact on students performance compared to rest of the students with an increase in class average of 3 points. In a survey conducted by in the United States under a project called Tomorrow [3] with 1.1 million students and 74,000 teachers in 2007, over 41% of students believe that on-line classes will have the greatest positive impact on their learning, a growth of over 20% from the 2006 data findings and over 26% of teachers chose on-line learning as their first choice for training. Researchers have suggested that m-learning enhances the collaborative and autonomous learning and it can be applied to a wide range of students without age boundaries, however, there are many obstacles exist in terms of implementing any significant m-learning applications.

The e-learning concept is well matured and there are many standard applications in use today. But the precise copy or transferring e-learning applications to m-learning will not satisfy learners in terms of usability or continuous learning. To make m-learning effective and useful in comparison with e-learning, there is a need for an understanding of the way in which the learner uses the mobile devices. For example, m-learners often utilize the system for learning usually for shorter periods unlike e-learners. Some of the major issues related to m-Learning can be listed as below:

- Limited user interface

[1] http://ojr.org/japan/wireless/1080854640.php
[2] http://www.learning2go.org/
[3] http://www.tomorrow.org

- Limited memory, battery life and storage of the mobile device are major inhibitors
- Irregular connectivity
- Cross-platform solutions, transmitting across different browsers and platforms are not yet widely possible
- The industrial closed proprietary solutions
- Existing applications are not easily integrated to all the mobile technology environment
- Continuous technology development militates against stability and sustainability in terms of mounting viable m-Learning applications

3.2 Requirements Analysis

M-Learning and e-Learning both are concerned with on-line learning. For example, the participants in an m-Learning as in an e-Learning environment are the learners, the author, the administrator and the tutor. Both provide the user with personalized learning experience. In both environments, a virtual tool of learning is required in order to allow a close follow-up of the training and management of the interactions between the various participants involved. But the m-learning system designers have to consider that even though mobile phones and computers possess very similar technological qualities, it carries different prerequisites.

3.2.1 Device Limitations

Considering present day scenario, there is a wide verity of devices available in the market with a range of technical configurations like PDA, Smart Phones, multimedia players, portable gaming consoles etc. All of these have some common constraints. Most of the devices have limited input capabilities like small number pad and several keys, even the devices with keyboard are small which are not convenient for an extensive usage. Therefore m-learning needed to be more click-centered unlike the type-centered e-learning. Screens are generally too small for the use of any sophisticated applications. Limited memory on the mobile device demands for a smaller, very well memory managed applications. Since the devices run on battery, the application needs to be "battery-friendly" as well.

3.2.2 Connectivity Limitations

This is a major barrier for developing an m-learning system. The bandwidth offered by the mobile networks is low for an extensive use of internet. Even with the introduction of high speed and high bandwidth services, internet usage in mobile phone is still very expensive compared to fixed line internet services. Another issue is existence of different communication networks around the world (e.g. GSM, GPRS, 3G and CDMA). The number of WLAN Hot-spots is increasing where people can access internet for free or for a fee. Therefore for an m-learner, it is important to have different cost effective possibilities to access the network from the device.

3.2.3 Institutional Limitations

Service provider is a vital part of any mobile phone. Since it is a business, according to their business model, restrictions are placed on the way the users can use their phones. For example, once a mobile phone is subscribed to a network service, the same device can not access other networks. Limitations are set not only the service provider, but also the phone manufacturer. The strict security and openness policies, iPhone does not allow running third party software. Even though the iPhone SDK is freely available and anyone can develop applications, to market those applications, developers has to get approval from Apple and approved softwares applications can be sold only using the Apple iTune store. Some manufactures also limit the size of the application which could be installed on the device.

3.2.4 Mobile Content Limitations

M-learning needs to consider the above mentioned limitations while preparing the learning contents. Creating unified solutions can be a challenge because the mobile industry is solidly divided and they have their own proprietary platforms and languages. Therefore the content author has to consider the target devices and their commonly supported presentation programs like web browser, so that the content can be automatically adaptable by the devices. Nowadays Wireless Access Protocol (WAP) 2.0 compatible browser on mobile devices uses standard HTTP over TCP/IP and displays web pages written in XHTML Mobile Profile (XHTMLMP) or WML [4] (Wireless Markup Language). (XHTMLMP is based on the eXtensible HyperText Markup Language (XHTML) [13] developed by the W3C to replace and enhance HTML and WML is a content formatting language specifically devised for small screen devices). The learning content can be found in lot more forms like images and multimedia which are not normally supported by the mobile devices. This demands for a content specific rendering method implementations on mobile devices. The content handled by the applications need to be organized into smaller units, so that it can be displayed in the small screen efficiently and easily readable for the learner. Since the connectivity could be a problem, there should be options for offline learning, by allowing learner to download the content to the device.

[4]http://www.wapforum.org

3.3 Related Work

There has been considerable research conducted by many institutions to address the technical issues faced while developing a mobile application. Standards for content that is intended to be delivered using web browsers installed on mobile devices are comprehensively advised by the activities of the W3C Mobile Web Initiative (MWI), a collaboration of industry and technology experts that is hoped to 'improve web content production and access for mobile users and the greater web.' Two main publications of the MWI are:

- Mobile Web Best Practices 1.0 [14], which specifies standards and best practices for delivery of mobile web-based content.
- W3C mobileOK Scheme 1.0. [15] It says, "mobileOK defines machine-readable content labels which may be applied to content to indicate that the content and its delivery pass a suite of tests based on the Mobile Web Best Practices document".

There has been no standard developed for mobile learning so far. But using the existing e-learning standards and models, m-learning applications have been developed.

Podcast

A podcast is a series of audio or video digital-media files which is distributed over the Internet by syndicated web feed. These web feed (Real Simple Syndication - RSS) is XML document which include web link to actual media files and meta-data such as authorship, publish dates, title, and descriptions the media file etc. The audio/video file and the feed are then posted to a Web server. The podcast creator must notify the audience of the existence of the podcast by publicizing the location of the feed. Many podcasters post a link to the RSS feed on their blogs, Web sites, or other public Web spaces. The podcast creator can also list information about the podcast in one of many directories that categorize podcasts alphabetically or by topic (e.g. iTunes). Podcast is distinguished from other digital-media formats by its ability to be syndicated, subscribed to, and downloaded automatically when new content is added. The listener is able to subscribe to the podcast series using a podcast aggregator (software that checks podcast feeds for updates at specified intervals) When the listener adds a new RSS feed, the aggregator downloads all episodes referenced in the current RSS feed. At regular intervals thereafter, the aggregator checks the feed for updates and downloads any episodes added since the previous check. Listeners can access podcasts directly on their computers, or on their portable MP3/video device. For those who prefer to listen on portable devices, most podcast aggregators will synchronize with portable devices automatically. Recording and distributing class lectures is considered one of the uses for podcasts. Podcasting is particularly well suited as a delivery mechanism for recorded classroom lectures because they occur multiple times a week over extended periods of time. The subscription model of file delivery affords maximum convenience for students, saving the steps of checking Web sites for multiple classes to download files several times a week. Stanford and many universities around the world now provide their audio and video learning content on iTunes 6 via iTunes Music Store.

Mobile Content

In a paper presented by Stuarts[16] a simple Rapid Content Development (RCD) method by which a practitioner, with limited technical knowledge and time, might produce learning objects for mobile phones by adapting existing materials. He used the MIMAS[5] hosted Hairdressing Training service as the source of these materials. The service offers guides on various hairstyles and aspects of the hairdressing industry for students. One guide was chosen as the object to be modified which consisted of 17 web pages of information and illustrations with images and video. The method adopted to create an object for a mobile environment is as follows:

- XHTML-MP combined with Cascading Style Sheets (CSS), which are used to code the web pages compatible for mobile device
- Java and XML based content creation tool called Maxdox, which is freely available for educational use
- Mobile movie format called Third Generation Partnership Project (3GPP)[6]

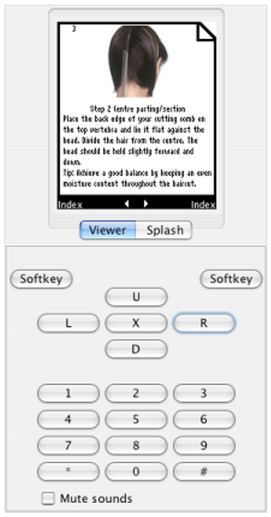
The RCD tool uses wizards to create the XHTML-MP based templates. Using those templates the learning content is created with option for simple linear navigation. The templates define the size and resolution of an image as well as media files. Once created objects can be downloaded and deployed via the web as .jar files on Java enabled mobile devices. This learning content can be played on the web browser easily as shown in figure 3.1.

Collaborative Learning

In a concept presented by Arrigo et. al. [17], they have developed a Java Environment for Learning Design (JELD) based on the IMS Learning Design specification. This framework consists of a core engine for running a learning process, a number of service modules and connection modules. The core engine receives an Extensible Markup Language (XML) Learning Design (LD) document as input and produces a run-time instance of a learning environment ready to be delivered to users. This process follows three main steps: validation, environment set-up and runtime creation. In the validation step, according to the

[5]http://www.mimas.ac.uk/
[6]

– http://www.3gpp.org/



Step 2 Centre parting/section
Place the back edge of your cutting comb on the top vertebra and lie it flat against the head. Divide the hair from the centre. The head should be held slightly forward and down.
Tip: Achieve a good balance by keeping an even moisture content throughout the haircut.

Figure 3.1: Mobile Content [16]

IMS guidelines, the core engine checks the references, semantics and completeness of the input document. Then a set of objects (Enterprise Java Beans) are instanced to set up the environment. Finally, the run-time environment is created, with the services, learning objects and user profiles defined in the input LD document. The service modules are used to provide the run time functions, e.g the the message exchange module is used to send e-mails and/or Short Message System (SMS) between users. Moreover, the interface of the learning run-time environment is adapted to the device using the connection modules. To over come the difficulty in making content available for use by several types of device, technologies used by JELD provide XHTML and XML Forms (XFORMS) interfaces to adapt information to the student client. Using XHTML ,the basic content is shared across desktops, PDAs and mobile phones.

Arrigo et. al. [17] explained a method to standardize collaborative learning using a mobile Collaborative Learning Tool (mCLT). The mCLT system is a Java™ Mobile Information Device Profile (MIDP) client for mobile telephones. It is characterized by a set of web services hosted on the server side and six mobile client modules. e.g Security Management module, Navigation module, Peer to Peer communication module and Local Repository module. Each module uses a Simple Object Access Protocol (SOAP) to interact with the web services. Navigation module manages a number of course activities like browsing the list of courses, peer -to peer module uses a message-system web service to send e-mails and Short Message System (SMS) between users.

Chapter 4

Mobile Application Development

There are a wide variety of mobile phones available in the market today, starting from a low end handset to iPhone with a beautifully designed touch screen. Added to it the devices come with a set of different applications. Based on the building architecture of the application on a mobile device, they can classified in to 3 basic types as below.

Browser Based Applications

A web browser is a common application available on most of the mobile devices now. Due to the mobile platform fragmentation that requires lot of different of versions of an application to support even a fraction of the market. Even web browsers are built on different application frame works, but they all usually connect to internet via a mobile service providers network, or via Wireless LAN (recently), using standard HTTP over TCP/IP and display web pages written in HTML, XHTML MP or WML. Therefore the mobile application developers looking to the browser to solve this problem when ever possible.

There are two separate models of web browser based mobile applications that are being discussed.

- One is widgets [18], which means download able applications which look and act like traditional applications, but are implemented using browser technologies including JavaScript, HTML and CSS. Widgets use APIs exposed either by the browser or by a widget engine (e.g Widsets, Zumobi, Plusmo) which needs J2ME.
- The other model is Asynchronous JavaScript and XML (AJAX) based Rich Internet Application (RIA) which is adapted in mobile application. [19] With this technique, JavaScript XMLHttpRequest object can communicate directly with the server without reloading the page. AJAX uses

asynchronous data transfer (HTTP requests) between the browser and the web server, allowing web pages to request small bits of information from the server instead of whole pages.

But both the techniques are not suitable when an application needs to work offline, like editing a document offline. They do not support any access to core phone features like the phone book, calendar, camera etc.There are few cross platform initiatives dedicated to solving these problems.

- Google's Gears[1] is one such project which enables a web application to work offline. Gears is currently available for Windows Mobile 5 and 6 devices as a plug-in for Internet Explorer Mobile. It introduced the LocalServer API and a database module which are key components that cache application's resources and make it available for offline use. When the offline application reconnects, any changes made in the local module synchronized with the server.

- Another example is the Open Mobile Terminal Platform's (OMTP) BONDI project [20]. BONDI defines interfaces and API's (a web run-time engine) that allow web based application access to device features(file system, camera, geolocation functionality) and user data (call log, phone book, calendar) in a standard, controlled and secure manner. Each web page containing JavaScript that accesses handset features through BONDI's web run-time engine will need to be signed with a special BONDI certificate. This certificate which will only be granted after the web service or widget has been tested and verified for security and functionality by an independent testing lab.

Platform Independent Players

Beyond telephony, the core services for most mobile carriers have centered on a suite of messaging applications. These applications include voice mail, SMS and MMS, video mail, and Voice SMS. Mobile devices are not only capable of handling the multimedia content from these messages, but also from other sources like internet or file transfer by the user . The underlying solution in mobile devices differs from device to device based on the platform and application programs available on the device.

For development of media applications, the native C++ application development kits provides the MMF (Multi Media Framework). The MMF client API implements several interfaces that encapsulate a lightweight plug-in framework for manipulation of audio and video features. For example: Audio playing, recording and conversion. This interface consists of three classes, CMdaAudioPlayerUtility, CMdaAudioRecorderUtility and CMdaAudioConvertUtility providing methods to create, play and manipulate audio data stored in files, descriptors and URLs. Audio streaming is supported by two classes CMdaAudioIn-

[1]http://gears.google.com/

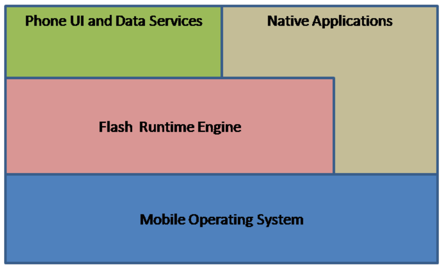


Figure 4.1: Flash Runtime Engine

putStream and CMdaAudioOutputStream which provide methods for recording and playing audio streams from a web address.

A Java Platform, called Java 2 Micro Edition[2] (J2ME), a version of the Java language that is optimized for mobile applications, is one of the the most widely available Runtime Environment for mobile devices. Two Java Specification Requests (JSRs) define the Mobile Services Architecture MSA : a full MSA stack that comprises 16 JSRs, and a subset of 8 JSRs. The MSA is an initiative that defines a common comprehensive cross- platform set of Java APIs, creating a predictable environment for portable Java applications aimed to minimize development costs and time. Games are the leading applications developed on J2ME, but it is also an important enabler in other application areas, such as personal productivity, messaging, location, and mapping. Mobile devices with J2ME come with Mobile Media API 1.1 (MMAPI), offers a range of multimedia capabilities for mobile devices, including playback and recording of audio and video data from a variety of sources. There are few features which are very important for today's and future application development which J2ME does not support. The first one being the support for accessing any type of databases (both internal or external) and the application developers have to use third party softwares like J2MEMicroDB [21] or other microDBMSs specifically created for mobile devices which use platforms such as Symbian or Windows Mobile. Even though J2ME supports a wide range of devices, applications developed on J2ME also need to use the device specific APIs for each type of device. So the original idea of "Write Once, Run Anywhere" does not really hold good in its true sense.

Adobe Flash Lite software is another powerful run-time engine for mobile devices which supports ActionScript 2.0. To run applications it needs a Flash Runtime Engine that runs on the mobile operating system as shown in the figure 4.1. This engine when integrated within a native mobile device application or embedded web browser, it supports applications and web content with FLV and SWF files. It also supports the World Wide Web Consortium's Standard Scalable Vector Graphics (SVG) Tiny, which is a mobile profile of the consortium's SVG recommendation.

[2]http://java.sun.com/javame/index.jsp

Native Applications

The native application implementation and deployment depends on the underlying hardware and the hardware abstraction layer of the mobile device and they can make use of all available features of the hardware like camera, network connectivity, audio and graphics. Some of the key advantages of developing a native applications are as follows:

- Access to the device feature : The system APIs facilitate the access to the device features like location, communication, media, and graphics capabilities.
- Performance : Due to their ability to interact with the system APIs directly application performance is high in terms of speed and reliability.
- Security: Native applications can make use of the security features supported by the device like locking the device and by nature the data stored with in the device is far more safer than on web application.

4.1 Application Development Platforms

In mobile devices, even if the applications have similar functionalities, they are not portable from one device to another, instead they need their own implementations. The design, development and implementation of application for mobile device is very different from application development for desktop PC due to the nature of the device and context of use. There are many mobile application development platforms in use today. They are different due to incompatible operating system, development models and communication options. Therefore, it is difficult to develop and deploy an application which is widely acceptable by many mobile devices.

4.1.1 Windows Mobile

The latest developer platform release from Microsoft[3] for the mobile devices is Windows Mobile (WM) 6.1[22]. Windows Mobile was developed to bring the PC experience to mobile devices and it is used in PDAs(Personal Digital Assistants) and Smart Phones (mobile phones with PDA features). The WM 6.1 software Development Kit (SDK) supports many tools and libraries that are need to develop mobile applications with Visual Studio 2005 as their Integrated Development Environment. It includes tools like the Device Emulator 2.0 which simplifies the process of developing applications that run across many different devices. The SDK includes support for .NET Compact Framework which is a subset of Microsoft's .NET Framework. It also includes recently released the Mobile Client Software Factory, which provides a framework and application blocks for commonly used smart client application scenarios. Even

[3]http://www.microsoft.com/22/en/us/default.aspx

mobile database-based applications can be developed using the SQL Server 2005 Compact Edition. It also has capacity to actively sync to the Exchange Servers, in general, windows mobile development environment is similar to the PC's, but with limited functionalities.

4.1.2 Symbian

Symbian OS is produced by Symbian Ltd., designed for mobile devices, with associated libraries, user interface frameworks and reference implementations of common tools. The Symbian Foundation, announced in June 2008 that, it will consolidate its different UI designing platforms (S60, UIQ, MOAP(S)) and make the platform open source. It has a IP networking architecture using which the mobile phones can have a smooth switching between the best connection types and networks (like WiMax and 3G) available at any point of time. The native graphics architecture in Symbian OS with a graphics composition engine enables application developers to make use of graphics hardware acceleration. It also provides transparency and overlays to the applications by running different applications as different processes parallelly, which gives users more context to applications, in addition to a 3-D experience. Since Symbian c++ (a variant of standard C++) is native to the device, developers can make use of all the hardware features of the mobile device while developing an application, but it also supports different run-time environments like Java and Flash. According to a survey conducted by Gartner, Inc[4].,[23] Symbian has the largest market share of about 50% as mobile operating system. Nokia, FOMA, Samsung and Sony Ericsson are some of the major manufactures use Symbian in their mobile devices.

4.1.3 Linux Based Platforms

LiMo

Limo[5] Platform is being developed by an industry consortium to enable the design, development and deployment of mobile phone devices based on a modular, middle-ware plug-in architecture (implemented in either C or C++) built around the open OS Linux. The middle-ware contains the security, network, messaging, multimedia, database frameworks. Additional frameworks can be added by the developer according to their needs in this architecture. Other than middle-ware, it has two components, the Application Manager Framework and the Application UI Framework. They are responsible for launching applications and defining the look and feel of the user interface. They ensure that an application that has focus receives user input and is able to render to the display. The Application Manager Framework also includes a secure package installer for downloaded applications. The kernel space contains the Linux kernel and device drivers and modem interface. Motorola, NEC, Panasonic, LG,

[4]http://www.gartner.com/
[5]http://www.limofoundation.org/

Vodafone are some of the manufactures use LiMo in their devices.

MOTOMAGX[6]

This is Motorola's open Mobile Linux platform with a support for third-party applications on Motorola devices. It supports three different application environments - Java ME, WebUI and native Linux providing developers choice in selecting the environment for their application needs. A special feature in this platform is the WebUI, which is based on WebKit(open source web browser engine), combines underlying Web 2.0 software technologies with access to local device resources, enabling developers to easily create personal and contextually aware Web 2.0 experiences for handsets.

4.1.4 iPhone

Apple[7] has released a SDK to the developer community for developing applications for iPhone. It includes Xcode (development environment with project management, a source editor, and a graphical debugger), iPhone Simulator and an UI Builder. Xcode has an application framework known as Cocoa which supports the programming language Objective-C 2.0. iPhone SDK allows developer to develop applications to use hardware and software features on the commercially available device like audio programming, the use of hardware sensors, multitouch, intercepting and handling event notifications for many iPhone-related events, raw video surfaces and 3D transformations.

4.1.5 Android

"Android is a software stack for mobile devices that includes an operating system, middle-ware and key applications. The Android SDK provides the tools and APIs necessary to begin developing applications on the Android platform using the Java programming language."[24]. It is a collaborative effort by the Open Handset Alliance which includes Google and more than 30 technology companies and telephone service providers. At the moment mobile operators control what applications and features operate on the handsets that use their networks which is not the case in PCs where user has almost complete control over it (both hardware and software). Google is trying to overcome this hurdle to some extent by making the platform open source and getting carriers around the globe involved in the Open Handset Alliance to give the user as much freedom as possible to run any compatible application they wish to use or allow them to develop their own application for the device.

The Android SDK (Android 1.0 SDK, Release 1) released by Google has a 4 level architecture and it includes an OS, a middle ware with core libraries and a run-time environment, an application development framework, and some applications as shown in the figure 4.2.

[6]http://developer.motorola.com/technologies/motomagx/
[7]http://www.apple.com/

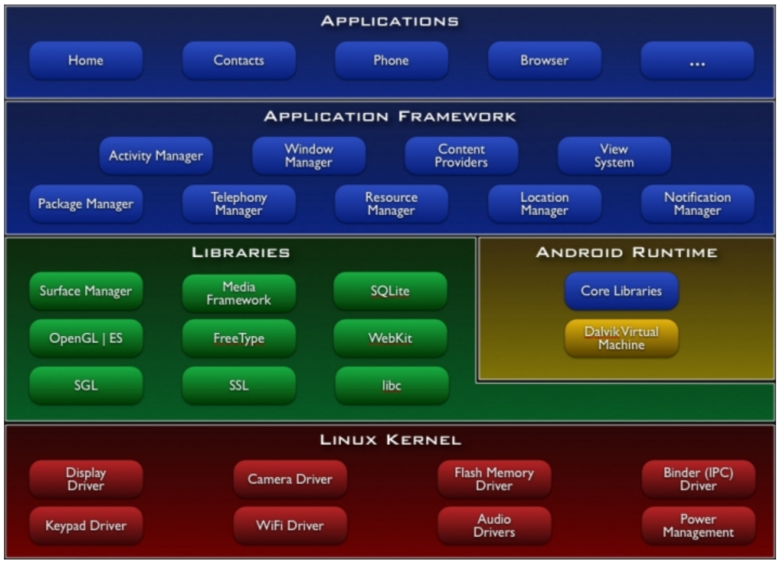


Figure 4.2: Android Architecture [24]

The first level contains the operating system which is a Linux 2.6 kernel with device drivers. This layer handles the security, memory management, process management, network stack, and driver model of the device. Next level has a set of c/c++ libraries and a Android Runtime engine. The libraries act as interface between the kernel/device drivers and application framework. Following are some of the libraries :

- SGL - 2D graphics engine
- Open GL - based on OpenGL ES 1.0 APIs; the libraries use either hardware 3D acceleration or highly optimized 3D software rasterizer
- Webkit - which powers both the Android browser and an embeddable web view
- SQLLite - structured data storage lightweight relational database engine available to all applications

Android includes a set of core libraries that provides most of the functionality available in the core libraries of the Java programming language. These libraries support the Android Runtime Engine called Dalvik Virtual Machine(DVM).The DVM relies on the Linux kernel for underlying functionality such as threading and low-level memory management. Every application on Android, runs as an independent process on its own instance of the DVM. The DVM executes files in the Dalvik Executable (.dex) format which is optimized for minimal memory footprint. A tool called “dx” is used to convert classes created by a Java compiler into the .dex format. The Application Framework layer is completely written in Java which is used by the application developed by user and the core applications alike. Another interesting feature of this architecture is that functions or resources published by one application can used by another application depending on the security constraints forced by the framework. The main components of this layer are as follows:

- Activity Manager - manages the life cycle of applications and provides a common navigation back stack for the applications running as a separate processes to have a smooth navigation.
- Package Manager - keeps track of the applications installed on the device.
- Window Manager - provides access to the core library called surface manager which manages application windows.
- Telephony Manager - provides the basic APIs for the telephone applications.
- Content providers - is a unique feature of this platform which enables an application to access data from other applications or to share their own data. For example, using content provider the data like phone number address from Contacts application can be shared by any other application deployed on the device.
- Resource Manager - provides access to non-code resources such as localized strings, graphics, and layout files.
- Location Manager - provides system with location services. For Example allow applications to obtain periodic updates of the device's geographical location or get an alert when the device enters the proximity of a given geographical location.
- View System - contains an extensible set of Views that can be used to build an application's UI like lists, grids, text boxes, buttons, and an embeddable web browser. It also handles the event dispatching, layout managing etc.
- Notification Manager - enables all applications to display custom alerts in the status bar.

The applications run on top of the Application Framework layer, which include both platform supplied and user developed applications. The SDK comes with a user friendly development environment including a device emulator, tools for debugging, memory and performance profiling, and a plug-in for the Eclipse IDE. The new user friendly desktop like application development approach makes it interesting to explore the possibilities on this platform.

4.2 Related Work

It has been more than 20 years since the mobile phone was introduced, but we did not see the same development as we saw in internet innovations. Even though there is a promising wide market, very few trivial applications developed over the years. According to Crowcroft [25], an inconsistent infrastructure, completely in-compatible handset Operating Systems, closed specifications etc leads to the absence of widely (device, platform independent) acceptable mobile applications. To overcome this division there are efforts made by people to tailor the same application to different devices.

Davis et. al. [26] in their generative approach, try to transform an application written on J2ME platform to a specific device with out manually rewriting the code. It was done using a specifically structured VoiceXML file as input to an XSL transformation. The transformation produces J2ME source code. Java servlets are used to compile the resulting code and package it with respect to a specific device.

The Mobile JSF (JavaServer Faces) Kit consists of the open source licensed MobileFaces core library to help Java EE developers to simplify the development of internet mobile applications. The Ericsson Mobile JSF Kit[8] provides a JSF library called MobileFaces. It contains a custom render kit to convert the content to mobile specific markup like XHTML MP / WML , a device manger to adapt any device specific needs, media adapter to handle the media content requirements of the mobile device, script renderer and content filter. MobileFaces abstracts the difficulties in the development of mobile applications, and enables rendering of different page content for different end devices based on one JSF page, there by reduces the amount of time needed for mobile application development, maintenance and extension.

An open source platform for developing .mobi-compliant applications called MyMobileWeb includes different modules which cover all the basic requirements that a complete and integrated mobile web site must fulfill, hiding from applications all complexity related to dealing with multiple delivery contexts. MyMobileWeb uses Wireless Universal Resource File (WURFL) to recognize and obtain the capabilities of devices. The WURFL is an XML configuration file which contains information about capabilities and features of many mobile devices. As a first step, the pages are defined in a declarative and in a device independent manner as XML files, and the user interface made up with abstract visual controls and containers.The layout and look and feel of visual controls is specified using W-CSS and custom extensions. Visual controls provide advanced user interaction and resolve common problems in the mobile environment, such as paging of long content. Based on the JSP (JavaServer Pages) 2.0 Expression Language , visual controls support data and content binding which is a technique that allows the interconnecting of data/content and presentation using declarative formalisms. This platform also provides an automatic validation framework that avoids dealing with different validation strategies depending

[8]http://www.ericsson.com/mobilityworld/sub/open/technologies/open_development_tips/tools/mobile_jsf_kit

on the delivery context. For those devices that support scripting at the client side, the scripting code is generated automatically. For devices without script support, validations are performed automatically at the server side.

Using an application platform called Breeze[9] developed by Cascada Mobile, one can build, test, and distribute mobile J2ME applications which can run on many mobile devices. The web application is written using HTML, CSS and JavaScript and then uploaded to a server where it gets converted into MIDlets (.jar and Java Application Descriptor(.jad) files) that will run across a wide range of devices. In addition, the application developers will receive a line of code they can put on their web sites, blogs, or social network profiles that let their visitors to download the application by entering their mobile number.

There are many researchers working on to establish guidelines for mobile application development. Häkkilä et. al. [27] discuss the challenges facing the design of context-aware applications and they propose 10 guidelines to address the problems. They are as follows:

- Consider the uncertainty in decision-making situations: While adaptating the content to the delivery context, there will be uncertainties resulting from context recognition and inferring logic. The designer must consider the level of uncertainty and decide to the inform the user before executing any actions.

- Prevention from interruptions: Designer must prioritize the actions for any possible interruptions during execution. e.g. a email notification or a calender alert.

- Personalization: Since mobile device is personal in nature, there could be settings related to individuals needs and preferences.

- Avoid information overflow: Especially during the actions like "push", the size of the information involved has to be considered due to the size of the device.

- Secure the user's privacy: While considering applications which share information with other device or the web, possibility to anonymity should be provided to avoid unwanted attention.

- Remember mobility: While the user is on the move, simple and fast interaction should be favored. e.g. location detection application

- Secure the user control: User should always have control over the device.

- Access to context: Sometimes it may be appropriate to provide the user a possibility to edit context attributes and their measures like letting the user rename locations or other context attributes can increase the understandability of the application.

[9]http://www.cascadamobile.com/products/breeze.php

- Visibility of system status: This is important in terms of knowing the system feedback for executed actions.e.g. In some case showing requests or other data related to information sharing may be relevant.
- Usefulness: Usefulness of the application in terms of usability, context adaptiveness, reliability in different situations or environments.

Since mobile application development is not yet as matured as PC application development, it might take some time to establish well accepted standards.

Chapter 5

SCORM Player Concepts

A SCORM learning environment needs a LMS and the client program, but there is often no Internet connection for a mobile device in many situations. So consequently there is no prospect to continuously connect to the LMS framework delivering the content. Therefore, there is a need for a method to substitute the navigation and content display ensured by the learning frameworks. With the use of a fabricated layer to replace the LMS, the content can be navigated back and forth according to the user's wish. This could be done with an offline SCORM player which uses the web browser object built into windows to display the SCOs with the help of a RTE within to respond appropriately to all the SCO - LMS communications.

5.1 Application Environment

The proposed SCORM Player's learning environment consists of the following components as shown in figure 5.1.

- SCORM Content package creator tool - which is capable of creating SCORM version 1.2 or version 2004 content packages.
- Storage – A simulated SD card which acts as a removable storage media for the emulator. The content packages are downloaded to this SD card and the SCORM player should be capable to play all the available content packages on the SD card.
- Data unzip utility - This component is a simple unzip utility which cycles through all of the entries in the file and copies each entry to a new file by reading its input stream. It unzips the archive file into the same directory and if an entry is a directory, it is created.
- Manifest Parser - To make use of the memory efficiently, this component must make use of a parsing technique which is suitable in resource limitations. The SAX specification can scan and parse large XML documents

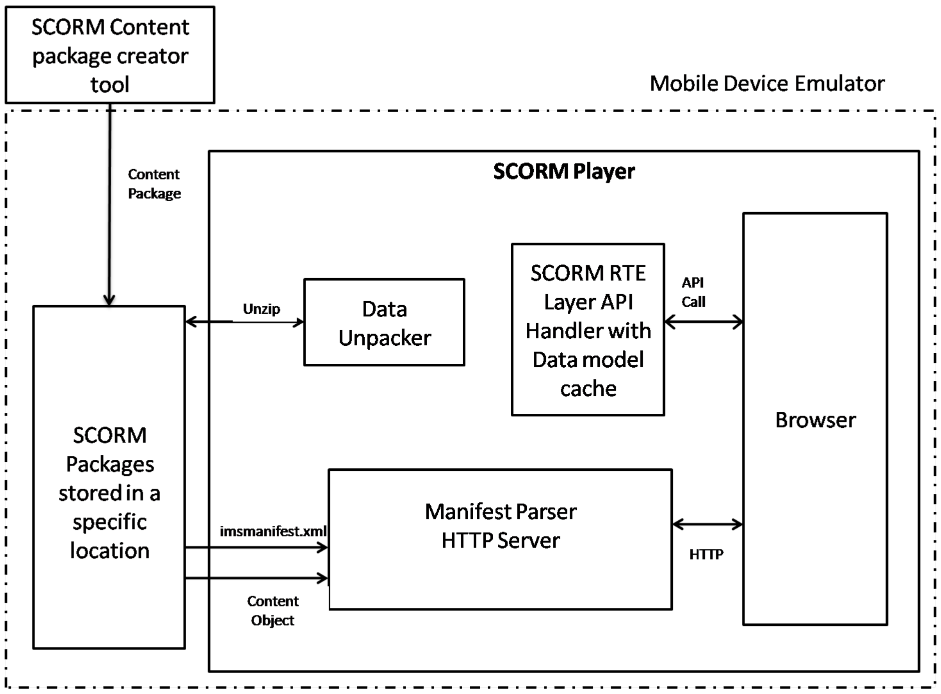

SCORM Content package creator tool
Mobile Device Emulator
Content Package
SCORM Player
Unzip
Data Unpacker
SCORM RTE Layer API Handler with Data model cache
API Call
Browser
SCORM Packages stored in a specific location
imsmanifest.xml
Manifest Parser HTTP Server
HTTP
Content Object

Figure 5.1: Application Environment

without hitting resource limits due to its event driven call back function approach. So SAX [1] Parser can be used to parse the imsmanifest.xml and create an instance of Organization which represents the activity tree of content package.

- HTTP server - A simple HTTP request processor thread to send the requested Asset or SCO to the SCORM player's browser.
- SCORM API handler and RTE Data model cache - This component is the actual replacement for the required LMS communications of the SCO. It should have the RTE data model implementation with the API interface for communication.
- Content browser - This component should allow the user to choose the content package and then allow to navigate through the content.

5.2 Architectural Pattern

The design of the SCORM player make use of the Model View Controller architectural design pattern[28]. In this pattern, Model contain the data of the application, as well as methods to access and manipulate the data. The View present the data to the user and the Controller processes the input and manages the model and view appropriately. In the following sections each of them with the coupling among the model, the views and the controllers are discussed. The MVC design of the SCORM player is shown in the figure 5.2.

Model

This application operates on 2 data models: The SCORM content model and SCORM RTE data model.

The conceptual representation (without attributes and method parameters) of the main classes of the content model is shown in the class diagram in the figure 5.3. The manifest file from the content package is parsed using SAX parser to create the content model classes for the further use by the controller functions.

During the learning experience, the RTE Data Model layer monitors the learners interaction with the content. In the SCORM player, to make the learning content navigation possible independent of a connection to a LMS, a RTE Data Model layer needs to be implemented. So that every RTE API method calls from the SCO are satisfied and an uninterrupted learning content navigation is achieved. The RTE Data Model uses a dot notation with every element name starting with "cmi". When a data model element is used in a SCO, they should be used as an ECMAScript characterstring and they should be in conformance with the Unicode Standard[11]. Even though the design of the SCO does not impose any mandatory conditions on using any API methods other

[1]http://www.saxproject.org/

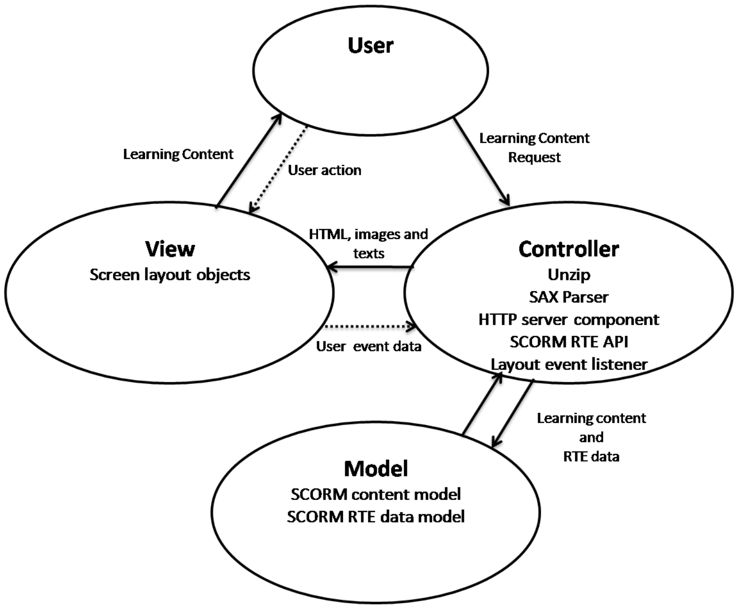


Figure 5.2: Model-View-Controller

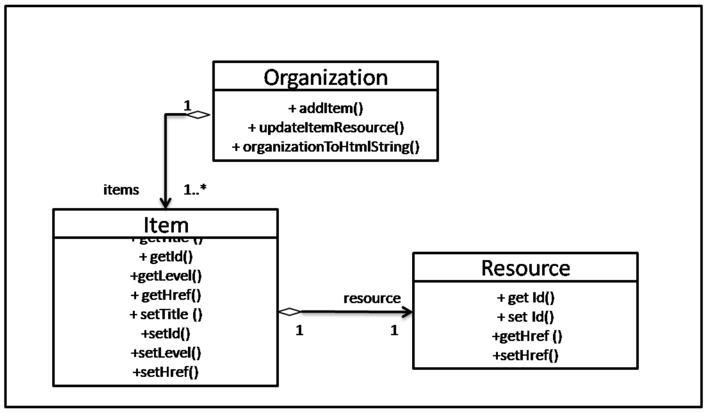


Figure 5.3: Content Model Class Diagram

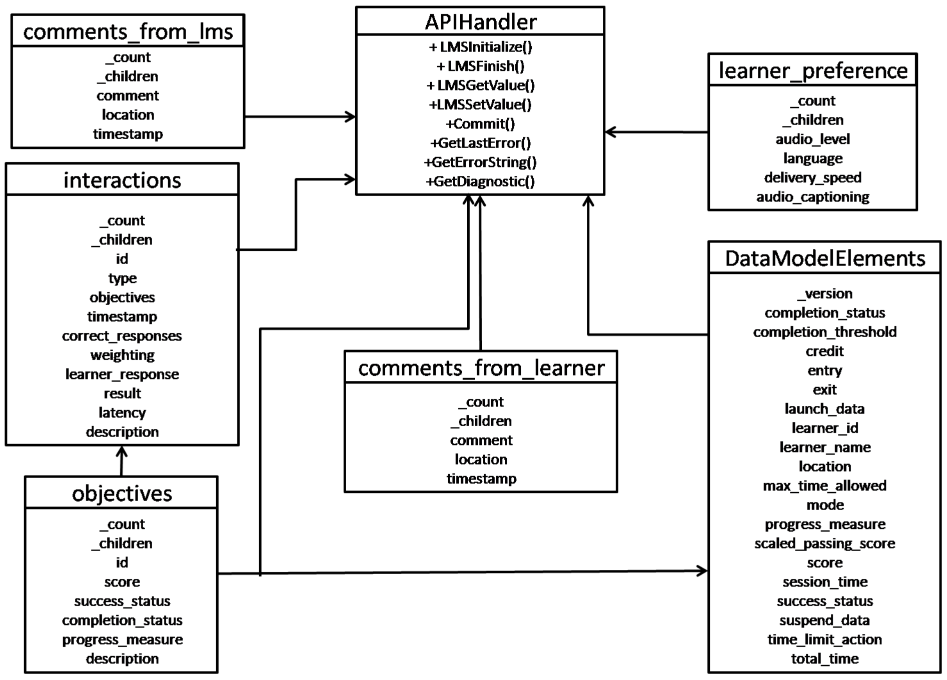


Figure 5.4: RTE Data Model

than initialize and finalize, hence the tracking becomes optional. Therefore the use of data model elements by SCO are optional too. But when they are used, they must follow the SCORM Data Model to ensure their compatibly and re-usability with different LMSs. However, a run-time environment must implement the complete data model elements in case a SCO needs to use any of them. The RTE data model consists of several data elements. Some of the data elements hold one record per SCO, while some can collect multiple of records with sub elements while tracking a SCO - learner interaction. The RTE data model for the SCORM Player is as shown in the figure 5.4. The API methods communicate with this data model's element classes via an API Handler, so that the presence of the LMS is imitated and the SCO can be navigated without the LMS. The life of tracking data communicated between the RTE data model and the SCO is one content package and the data model is refreshed by invoking every next content package. Therefore data are not stored in any persistence storage but it is stored within the application's data model class.

View

The complete learning experience of a learning object is realized by using three screens in this SCORM player as shown in figure 5.5. All the three screens are designed as display only screens with selectable list and buttons to navigate from

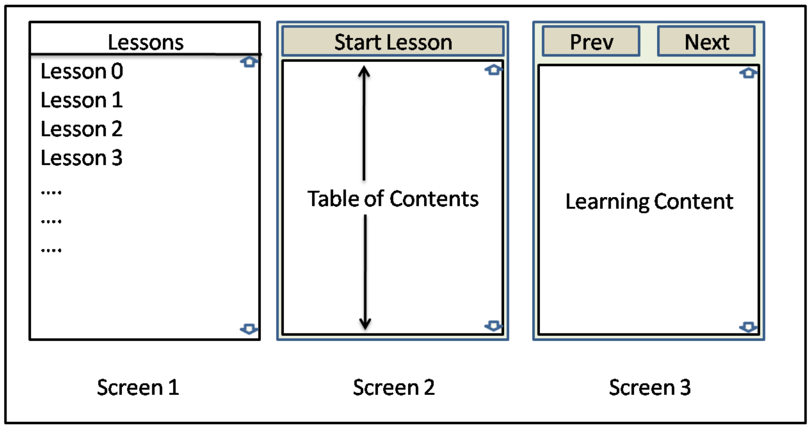


Figure 5.5: User Interfaces

screen to screen. The SCORM player can play only one content object at a time with an option to select any SCORM content package (Lesson) available within the device from the first screen. The screen two shows the content package in the form of table of content in a plain text. The button "Start Lesson" takes the user to the last screen where the actual content of the learning object can be navigated sequentially in both forward and backward directions. The navigation is facilitated by the use of two buttons "Next" and "Prev". This player is capable of presenting learning contents with plain text, HTML and images (.PNG and .JPG).

SCORM player must present all the available content packages to the user at one glance at the start of the application. List view represents a common interface navigation metaphor for mobile devices. It can be used to present multiple simultaneous navigation choices for relatively screen sizes or large screens with touch screens. Therefore, a List view can present all the available content packages as different lessons. In the content area of screen 2 and 3, the contents of the objects displayed. If the content is larger than the visible range, a scroll bar can be used, with which the user view through the entire content. In this approach, the layout remains constant irrespective of the type or size of the content.

Controller

The controller of this application comprises of the following components.

- The SCORM CP come in archive file formats, therefore each content package is unzipped.
- For every package present in the device, an activity tree has to be built for the navigation. The SCORM 1.2 packages do not have any sequencing information within the CP, but the SCORM 2004 has the sequencing rules

and conditions within the CP. The SCORM player is designed to navigate through the entire content and therefore the sequencing information are ignored while building the activity tree from the manifest by parsing the imsmanifest.xml file.

- The content packages are presented as a list of lessons to the user. Each lesson is linked to an activity tree.
- As a request for a lesson is received in the form of a HTTP request, selected content package's first leaf (Asset or SCO) is sent to the browser as a HTTP response. Similarly successive content requests are fulfilled. This component is a HTTP server which starts with the application. It has a socket that is bound to a specific port number. The server waits, listening to the socket for a HTTP request and when there is a request, the server accepts the request. Upon acceptance, the server gets a new socket bound to the same local port and also has its remote endpoint set to the address and port of the browser client. Once the request is processed, the response is sent back to the browser. It needs a new socket so that it can continue to listen to the original socket for connection requests while tending to the needs of the connected client. This server component must run as a separate thread, so that the server can wait for the requests independent of the other activities of the SCORM Player.
- The launched SCO interacts with the RTE data model via the API calls. Here the RTE data is not stored; therefore the data from the previously viewed SCO does not influence the successive SCO navigation.

5.3 Network Access Behavior

The Android SDK supplied mobile device emulator allows prototype, develop, and test Android applications without using a physical device. The hardware features emulated by this emulator are the following:

- An ARMv5 CPU and the corresponding memory-management unit (MMU)
- A 16-bit LCD display
- A Qwerty-based keyboard and associated Phone buttons
- A sound chip with output and input capabilities
- Flash memory partitions (emulated through disk image files on the development machine)
- A GSM modem, including a simulated SIM Card

Each emulator instance runs behind a virtual router, but unlike an actual device connected to a physical router, the emulated device doesn't have access to a physical network. Instead it runs as part of a normal application on the development machine. Any application running on the emulator uses the virtual router for the network connection and the virtual router is able to handle all outbound TCP and UDP connections/messages on behalf of the emulated device, provided the development machine's network environment allows it to do so. There are 3 possible scenarios for accessing the SCORM learning contents from the emulator for online and offline learning. These scenarios remain valid for a real mobile device as well, because the application itself has no role in establishing or maintaining the network connection.

- When the device is connected to internet, the web browser can be used to directly connect to the LMS and to play the SCORM contents. Under this condition in fact the device needs to have a continuous internet connection. The web browser has its limitations with respect to handling the framed layouts, JavaScript support, content encoding and mime types. At present the SCORM contents available from most of the content providers are meant for the PC users, so they cannot be fully functional on the Android web browser.

- When the device has a limited internet connection, the SCORM package can be downloaded to a predefined location (e.g. in the android emulator SD card can be accessed with an absolute path "/sdcard/") in the device and the SCORM player can play the learning contents from that location.

- If the device is offline, SCORM packages can be downloaded to a memory card using a PC with internet connection. That memory card can be plugged on to the mobile device and the SCORM player can then play the contents. The device emulator uses mountable, writable disk images stored on the development machine to simulate removable flash (e.g. SD card) partitions on an actual device. The Android SDK provides tools to make the disk images and to copy files to the disk image.

Chapter 6

Implementation and Testing

The SCORM player developed on the Android platform during the course of this thesis is capable of navigating SCOs and Assets without connecting to a LMS. The students may download the materials complying with SCORM 1.2 or 2004 and the content can be displayed offline in a content browser. The SCORM player is an independent application on the android platform. This application like any other application developed for Android is placed on the Application layer. The applications installed on application layer appear on the home screen of the android emulator with the native applications as shown in figure 6.1.

The SCORM Player is developed using java language with android specific java libraries. Eclipse is used as the development tool with a Google supplied android plug-in and Android 1.0 SDK Release 1.

6.1 Implementation

This application is built using one of the building blocks of the Android application development called Activity. Activity base class is extended and used to display a single screen, handle the user interactions with that screen in the application. Each of the three screens screen1, screen2 and screen3 corresponds to three classes SCORMPlayer, LessonLauncher and ContentViewer respectively.

An Android Activity class must implement the onCreate() method which initializes the activity. In this method by calling setContentView(int) with a layout resource defining UI of the activity the screen is displayed and the method findViewById(int) is used to retrieve the widgets in that UI, so that the user interaction can be managed within the program. After implementing the Activity class, to use it in the application, activity class must have a corresponding <activity> declaration in the application's manifest file. The Android manifest file is a mandatory file for every application in Android and it is named as AndroidManifest.xml. It describes global values for the package, including all the application components (like activities, services, etc.) that the package exposes, the implementation classes for each component and where they can be

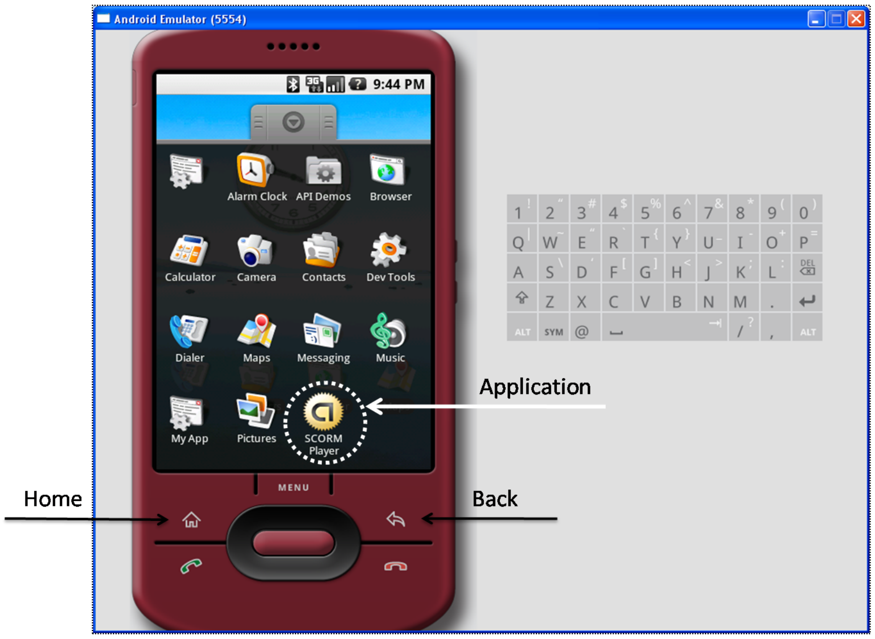


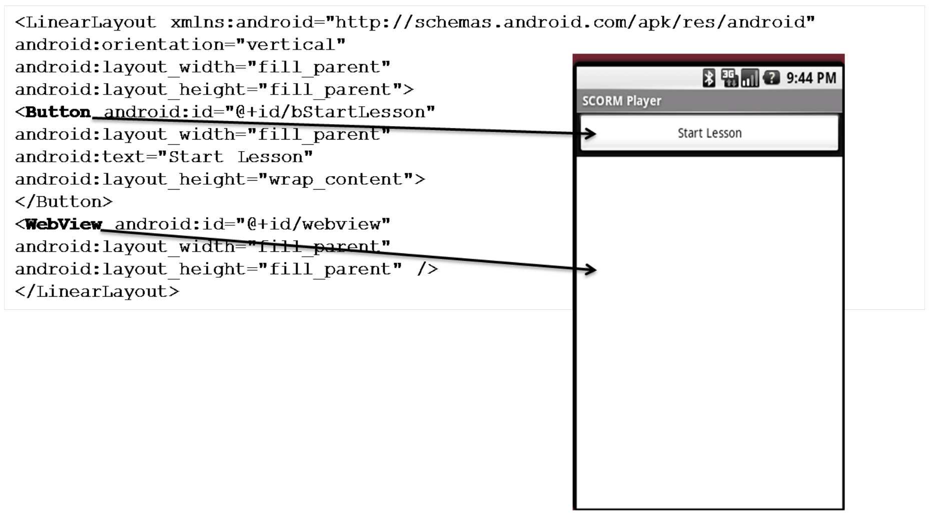
Figure 6.1: Emulator

launched.

Android supports two types of User Interface (UI) design methods. They are either using program code or using XML file. Android defines a large number of custom elements, each representing a specific Android UI subclass. With the second method, UI can be designed the same way as a HTML file with a series of nested tags, saved in an XML file. Each file describes either a simple visual element, or a layout element that contains a collection of child objects (a screen or a portion of a screen). When Android SDK compiles the application, it compiles each file into a UI resource that can be rendered on the screen by the application's controller classes. In our application, all the screen layout and screen components are designed using XML. Each screen is defined using a tree of Views, which are the basic unit of UI design in Android. A View occupies a rectangular area on the screen and it is responsible for drawing and event handling. The SCORM player's UI uses four different UI components; they are ListView, WebView, ImageView and Button.

A ListView shows selectable items in a vertically scrolling list. An Adapter holds the entire list's underlying data as well as the Views necessary to render each row of the ListView. An Adapter object acts as a bridge between a View and the underlying data for that View. The Adapter can get data from an array of arbitrary objects like list or a database table cursor. On an operation on the list item, adapter object provides access to the underlying data items.

To render web content, the mobile devices come with native web browsers. In



```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:orientation="vertical"
android:layout_width="fill_parent"
android:layout_height="fill_parent">
<Button android:id="@+id/bStartLesson"
android:layout_width="fill_parent"
android:text="Start Lesson"
android:layout_height="wrap_content">
</Button>
<WebView android:id="@+id/webview"
android:layout_width="fill_parent"
android:layout_height="fill_parent" />
</LinearLayout>
```

Figure 6.2: layout_web.xml

Android, the web browser can be made part of a UI with other UI elements. The WebView is a class belongs to the WebKit [1]web browser engine used to render web contents to a screen. It is based on the WebCore[29] libraries and DOM [30] libraries for layout, HTML rendering and JavaScriptCore[29] for JavaScript interpretation. WebView is used on the screen 2 and screen 3 to display table of contents and learning content. In XML the WebView layout definition of screen 2 appears as shown in figure 6.2. Here the <LinearLayout> means the UI components with in this layout will be arranged linearly. The height and width attributes with value "fill_parent", makes the layout to stretch over the entire screen area. Button and the WebView are placed inside the LinearLayout, so that they are placed one after the other.

The application makes use of this screen layout files as shown below. The method setContentView of the Android Activity sets the layout defined in layout_web.xml as its current UI which results in the screen as shown in figure 6.2.

```
WebView myLessonView;
setContentView(R.layout.layout_web);
myLessonView = (WebView) findViewById(R.id.webview);
```

The ImageView component can be used within any layouts to render images on to the screen. Due to the limitation of the WebView to render contents only up to 8KB, as and when the content is an image Asset, ImageView component replaces the WebView on Screen 3. This view also allows the scaling of images

[1]http://webkit.org/

```
<ImageView android:id="@+id/imageview"
android:layout_width="fill_parent"
android:layout_height="500px"
android:layout_x="0px" android:layout_y="50px"
android:adjustViewBounds = "true"
android:scaleType="fitStart"/>
```

Figure 6.3: ImageView

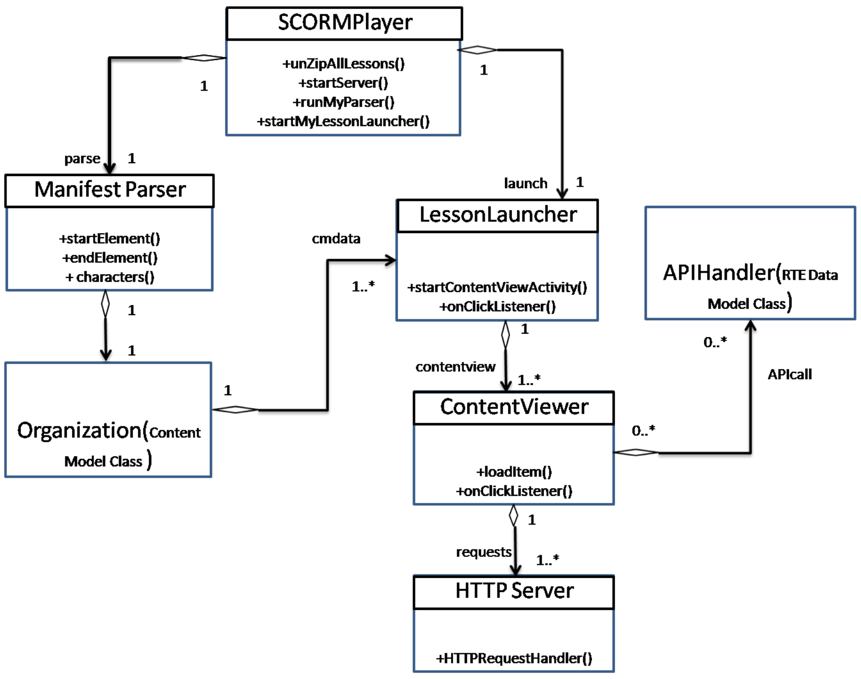


Figure 6.4: Class Diagram

(PNG and JPEG), so that the image can be scaled to fit the screen layout of the device. The ImageView is defined as shown in figure 6.3.

In the figure 6.4 the main classes and the main methods used in this application are shown. The attributes and parameter passed on to the methods are omitted in this class diagram.

- SCORMPlayer : This is the starting class of the application. At the beginning of this activity, each content package is unzipped and the activity tree is built. After unzipping the content packages, a list is made with a link to each activity tree. This list is rendered as a selectable list on to the screen 1. A HTTP server thread also starts to run as a result of activation of this class. Onclick event on any of the list item activates the LessonLauncher activity.

- LessonLauncher: This class displays the Table of Contents from using the content model class Organization. On click of “Start Lesson” button launches ContentViewer activity for the selected content package.

- ContentViewer : In this Android Activity’s UI , based on the type of the content to be displayed either WebView or ImageView are selected. This is implemented by placing two views at the same position on the learning content area. The implementation is done as shown in the following code. If the SCO or Asset type is .html or .htm, the content data is loaded to the WebView. Otherwise, if the content type is .jpg or .png, ImageView is brought to the front, the image files are decoded to Bitmaps and displayed.

```
if (path.endsWith(".html") || path.endsWith(".htm"))
    {
  byte[] array = new byte[(int) length];
  try {
        InputStream is = new FileInputStream(f);
        is.read(array);
  }catch (FileNotFoundException ex) {
        return;
  } catch (IOException ex) {
        Log.v(TAG, "Failed to access file: " + path);
  }
String data = new String(array);
myLessonView.bringToFront();
myLessonView.loadData(data, "text/html", null);
}
if (path.endsWith(".jpg")|| path.endsWith(".png")) {
try{
   myImageView.bringToFront();
                                    myImageView.
       setImageBitmap(BitmapFactory.decodeFile(path));
}
```

  To enable a connection between the APIHandler class and the API instance on the WebView instance of the ContentViewer, Android allows JavaScript binding for the WebView. It is done in 2 steps as shown below.

```
WebView myLessonView;
WebSettings myWebSettings;
myWebSettings = myLessonView.getSettings();
myWebSettings.setJavaScriptEnabled(true);
myLessonView.setWebChromeClient(new WebChromeClient()
    );
myLessonView.addJavascriptInterface(new APIHandler(),
     "API_1484_11");
```

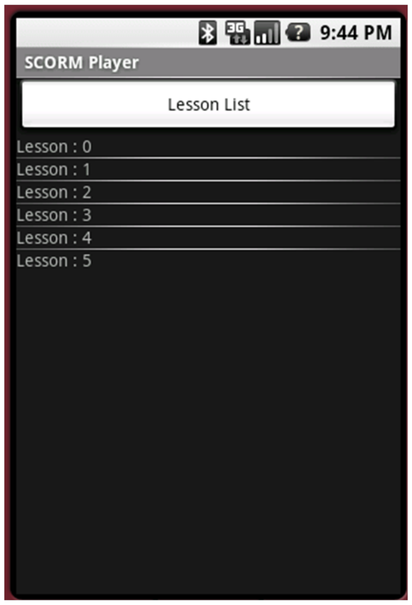


Figure 6.5: Screen 1

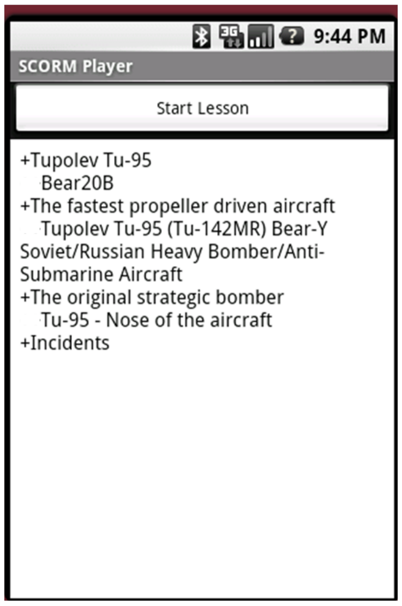
> At first, the JavaScript enabled WebView has to be set as the Android version of Chrome Browser. To enable a bridge between the APIHandler class and the launced SCO on the browser, the WebView component has to have Java Class binding. i.e, the APIHandler class should be bound to the WebView with a name "API_1484_11". With this, the SCO will be able to communicate to the APIHandler class using the name "API_1484_11". Now the API calls like GetValue or SetValue will invoke the functions in the RTE data model class APIHandler which in turn operate on the data model elements.

The main difference between these two versions (SCORM 1.2 and 2004 versions.) is the conditional sequencing information present in the manifest file of version 2004. The SCORM Player ignores the conditional sequencing information in manifest at the parsing stage, and rest of the functionalities remain same for both. Therefore, user can browse through the contents from both versions.

6.2 Testing

Functionality Testing

As we start the SCORM Player, all content packages stored in the mobile device emulator's SD card appears as a selectable list. The Player gives each of the packages a name as "Lesson:" with an index number as shown in figure 6.5. A


9:44 PM

SCORM Player

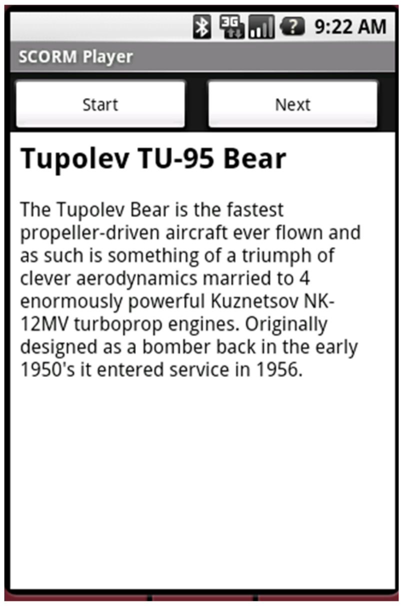
+Tupolev Tu-95
  Bear20B
+The fastest propeller driven aircraft
  Tupolev Tu-95 (Tu-142MR) Bear-Y Soviet/Russian Heavy Bomber/Anti-Submarine Aircraft
+The original strategic bomber
  Tu-95 - Nose of the aircraft
+Incidents

Figure 6.6: Screen 2

lesson selection and click action by the user, activates screen 2 as shown in figure 6.6. It displays all the resources (SCO or Assets) present with in that content package in the form of a Table of Contents with a “Start Lesson” button on top of the screen layout. Upon pressing the “Start Lesson”, screen 3 is displayed as shown in figure 6.7. It has two buttons on top of the screen namely “Start” and “Next”. The lower part of the screen displays the content of the first item on the Table of Contents. As the “Next” button is pressed, the next item from the Table of Content gets displayed and the button text changes to “Prev” and “Next”. At the end of the content Navigation the top button texts change to “Prev” and “End”. If the content is a image, it appears as shown in figure 6.8.

Results

- This SCORM Player application is able to play SCORM version 1.2 and 2004 content packages successfully.
- If the content is HTML and too long to fit in the scree at one glance, the scrolling bars gets activated and the entire content can be viewed by scrolling up and down.
- The player’s browser is unable to play HTML content of size more than 8KB due to the restriction in the current SDK.
- The content browser does not display anything, if the content is anything other than simple HTML files or image files (.png or .jpeg), but it will



Tupolev TU-95 Bear

The Tupolev Bear is the fastest propeller-driven aircraft ever flown and as such is something of a triumph of clever aerodynamics married to 4 enormously powerful Kuznetsov NK-12MV turboprop engines. Originally designed as a bomber back in the early 1950's it entered service in 1956.

Figure 6.7: Screen 3a

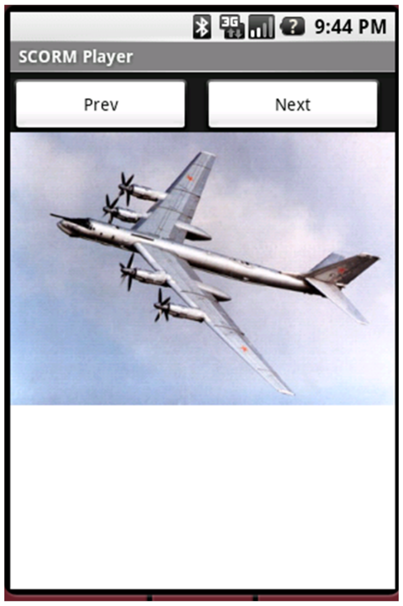


Figure 6.8: Screen 3b

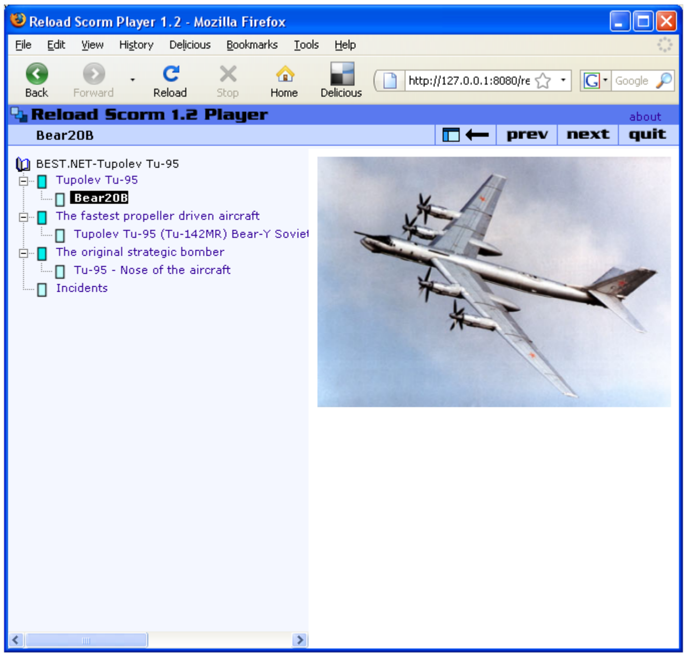

Reload Scorm Player 1.2 - Mozilla Firefox
File Edit View History Delicious Bookmarks Tools Help
Back Forward Reload Stop Home Delicious
http://127.0.0.1:8080/re
Google
Reload Scorm 1.2 Player
about
Bear20B
prev next quit
BEST.NET-Tupolev Tu-95
Tupolev Tu-95
Bear20B
The fastest propeller driven aircraft
Tupolev Tu-95 (Tu-142MR) Bear-Y Soviet
The original strategic bomber
Tu-95 - Nose of the aircraft
Incidents

Figure 6.9: Reload SCORM Player

allow the user to continue navigation to the next item.

- In order to compare the results with respect to a available PC application, a SCORM player called Reload[2] SCORM player 1.2 (plays SCORM 1.2 contents) has been chosen. The content package used in functionality testing is played using the Reload SCORM Player as shown in figure 6.9. The screen 2 corresponds to the Left side panel of the Reload player. The Screen 3 corresponds to the right side panel of the Reload Player. During this comparison, we found that, there is no difference between the navigation of the contents between the two players.

Evaluation

- The application is tested for the linear navigation, HTML and image (.PNG and .JPEG) contents. In both cases, the results meet the expectation. But the application does not handle any other file types due to the limitation on the current version of WebView and ImageView UI controls.

- The UI design is purpose focused and intuitive for the user with minimal user input requirement.

[2]http://www.reload.ac.uk/

- The results obtained while playing both versions 1.2 and 2004 are as expected.
- The SCORM Player plays one content package at a time and the RTE data model is not stored nor available for the user.
- The content packages need to be in the SD card for the application to locate them.
- All the content packages tested were authored for PC users. Therefore few packages were not successfully playable in the SCORM Player. This is due to the fact that, the web browser component in Android is a mobile version and it lacks support for JavaScript, image files, HTML frames, CSS etc. unlike PC web browsers.
- Start up time of the application depends on the number of SCORM content packages available on the device, because unzip and parsing are done during the startup.
- The responsiveness of the application to a request is immediate, so the user does not experience any delay.

Chapter 7

Conclusion

In this thesis a SCORM player is designed, implemented and tested on the mobile device platform Android. It is done in following steps with a pre-requisite of downloading SCORM content package archive files on to a preset location in the device:

- Unpacking of the content package
- Parsing the content manifest to build an activity tree for each of the content package available on the device and all the content packages are made available for the user's selection
- Content browsing on a viewer with the aid of a HTTP server component for the content resource launch
- SCO's communication with RTE data model via an API

This SCORM player facilitates offline sequential navigation of SCORM content on mobile devices. It is expected to run on any mobile device which supports Android platform, which is expected to be adapted by different mobile device manufactures, since the libraries used to develop this application will remain the same in all devices. This application does not need any special setup requirements and it can be made functional by installing one executable file (.apk) into an appropriate directory in the device.

Issues which could be further explored as part of future work:

- Depending on the future versions of the platform, enable support for different file types like Flash and multimedia files.
- Add additional functionality to enable the player's runtime environment to work directly with a remote LMS.
- This application is developed using the SDK on a smart phone device emulator, the implications of running this application on a real mobile device is still unknown. So deploying it on device could result in varying outcomes which needs to be addressed.

Bibliography

[1] Reuters. Global cell phone use at 50 percent. http://www.reuters.com/article/technologyNews/idUSL2917209520071129, November 2007.

[2] The Times of India. 100 million new mobile users in 1 year. http://timesofindia.indiatimes.com/100_million_new_mobile_users_in_1_year/rssarticleshow/3455056.cms, September 2008.

[3] Jill Attewell. *A technology update and m-learning project summary*. Learning and Skills Development Agency, London, UK, 2005. http://www.m-learning.org/knowledge-centre/m-learning-research.htm.

[4] Hansjörg Von Brevern. Cognitive and logical rationales for e-learning objects. *Educational Technology & Society*, 7(4):2–25, 2004.

[5] Advanced Distributed Learning Initiative. *Sharable Content Object Reference Model (SCORM®) 2004 3rd Edition, Run-Time Environment Version 1.0, 2006, Sequencing and Navigation Version 1.0, 2006. Content Aggregation Model Version 1.0, 2006, Overview SCORM® 2004 3rd EDITION*, 3 edition, November 2006. www.adlnet.gov.

[6] Carol Fallon and Sharon Brown. *E-learning Standards: A Guide to Purchasing, Developing, and Deploying Standards-conformant E-learning*. CRC Press, Routledge, USA, 2002.

[7] AICC. *DOCUMENT NO. CMI001 , AICC CMI Guidelines for Interoperability*, August 2004. http://www.aicc.org/pages/down-docs-index.htm.

[8] IMS Global Learning Consortium. *IMS Content Packaging Specification v 1.2*, 2005. http://www.imsglobal.org/specifications.html.

[9] IEEE Learning Technology Standards Committee. *Standard for Information Technology - Education and Training Systems - Learning Objects and Metadata*, ieee 1484.12.1-2002 edition, 2002.

[10] IMS Global Learning Consortium. *Standard for Learning Technology-ECMAScript Application Programming Interface for Content to Runtime Services Communication*, ieee 1484.11.2-2003 edition, 2003.

[11] The unicode standard, version 4.0.0, 2003. The Unicode Consortium.

[12] ADL Technical. The scorm 2004 3rd edition conformance test suite version 1.0.2. http://www.adlnet.gov/downloads/downloadPage.aspx?id=277.

[13] Xhtml 1.0 the extensible hypertext markup language (second edition). W3C Recommendation, 2002. The World Wide Web Consortium.

[14] Mobile web best practices 1.0. W3C Recommendation, 2008. The World Wide Web Consortium.

[15] W3c mobileok scheme 1.0. W3C Working Draft, 2008. The World Wide Web Consortium.

[16] Stuart Smith. Mobile learning. *Association for learning technology*, 7, January 2007. http://newsletter.alt.ac.uk/e_article000729140.cfm.

[17] Jill Attewell and Carol Savill-Smith. *Learning with Mobile Devices - A Book of Papers*. Learning and Skills Development Agency, London, UK, 2004.

[18] Widgets 1.0 requirements. W3C Working Draft, 2008. The World Wide Web Consortium.

[19] The World Wide Web Consortium and Open Ajax Alliance. *Position Papers on Mobile Ajax*, September 2007. http://www.w3.org/2007/06/mobile-ajax/papers/.

[20] Open Mobile Terminal Platform Team. Bondi architecture & security requirements public working draft v0.43. http://www.omtp.org/Bondi/PublicDraft.aspx, August 2008.

[21] Marc Alier, Pablo Casado, and Ma José Casany. J2memicrodb: a new open source lightweight database engine for j2me mobile devices. In *MUE '07: Proceedings of the 2007 International Conference on Multimedia and Ubiquitous Engineering*, pages 247–252, Washington, DC, USA, 2007. IEEE Computer Society.

[22] Microsoft. Architectural overview of windows mobile infrastructure components. http://www.microsoft.com/windowsmobile/en-us/business/solutions/enterprise/architectural-overview.mspx, April 2008.

[23] Gartner Inc. Gartner says worldwide smartphone sales grew 16 per cent in second quarter of 2008. http://www.gartner.com/it/page.jsp?id=754112, September 2008.

[24] Google Technical Team. Android. http://code.google.com/android/, 2008.

[25] Jon Crowcroft. New directions in mobile communications, or how to learn to stop hating the cellular telephone industry. *SIGCOMM Comput. Commun. Rev.*, 38(2):51–53, 2008.

[26] Victoria Davis, Jeff Gray, and Joel Jones. Generative approaches for application tailoring of mobile devices. In *ACM-SE 43: Proceedings of the 43rd annual Southeast regional conference*, pages 237–241, New York, NY, USA, 2005. ACM.

[27] Jonna Häkkilä and Jani Mäntyjärvi. Developing design guidelines for context-aware mobile applications. In *Mobility '06: Proceedings of the 3rd international conference on Mobile technology, applications & systems*, page 24, New York, NY, USA, 2006. ACM.

[28] Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, and Michael Stal. *Pattern-Oriented Software Architecture Volume 1: A System of Patterns.* John Wiley and Sons Ltd, Chichester, UK, 1996.

[29] WebKit Team. The webkit open source project. http://webkit.org/, 2008.

[30] Document Object Model Working Group. Document object model level 3 recommendations. http://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407/, April 2004.